MorphoDig User's guide
MorphoDig v1.1

Renaud LEBRUN

Institut des Sciences de l'Evolution, University of Montpellier, France

October 12, 2018

Contents

Introduction

Contents

MorphoDig is based on the design concepts of the software FoRM-IT (Fossil Reconstruction and Morphometry Interactive Toolkit; [Zollikofer and Ponce de León, 1995, 2005]). MorphoDig[Lebrun, 2018] was developed as a help to the scientific journal MorphoMuseuM (M3), in order to help scientists to produce enriched surface models. The source code is hosted on Github.

Origin of the project

Over the last two decades, even though 3D data acquisition and computer-assisted techniques have grown increasingly popular among biologists, paleontologists and paleoanthropologists, so far, no standard biology-oriented 3D mesh manipulation software has emerged; most of the time, researchers either use commercial software which are not primarily designed for biologists, or develop their own in-house software solutions. MorphoDig builds upon the design concepts of the software FoRM-IT (Fossil Reconstruction and Morphometry Interactive Toolkit; [Zollikofer and Ponce de León, 1995, 2005]), and is developed to meet the need to ease the production and the diffusion of 3D models of biological organisms. MorphoDig provides a set of tools for editing, positioning, deforming, labeling, tagging sets of 3D surfaces. As such, MorphoDig can be used to produce enriched models which can in turn be submitted to M3.

Main features

Features include:

- Retro-deformation for virtual restoration of fossils/deformed specimens;

- Point and curve primitives for placing the exact type of landmark points you're interested in
- Easy to use 3D interface for positioning and manipulating sets of surfaces and landmark primitives
- Mesh tagging, labeling and coloring (to allow for the creation of anatomy atlases)
- Mesh scalar computation and coloring (based upon curvature/thickness etc...)

MorphoDig allows to import and export 3D meshes in standard formats such as STL, PLY and VTK polydata, and as such, in can be used in conjunction with a variety of other 3D mesh editors such as MeshLab (http://meshlab.sourceforge.net/) or Blender (https://www.blender.org/)

Implementation

MorphoDig is entirely written in C++, and uses the visualization library VTK [Avila et al., 2001]. The GUI has been designed with QT (https://www.qt.io/). MorphoDig is open-source and cross platform, and we are looking forward to welcoming new developers in the future in order to implement new functionalities.

Chapter 1

Licence

Contents

1.1 MorphoDig

1.2 VTK

Chapter 2

F.A.Q.

Contents

2.1 How should I cite MorphoDig in scientific publications ?

You may cite MorphoDig with the following reference :
Lebrun, R. MorphoDig, an open-source 3D freeware dedicated to biology. IPC5, Paris, France; 07/2018.

2.2 Is MorphoDig a geometric morphometrics software ?

No. However, you can digitize 3D landmarks on complex 3D surfaces using MorphoDig, which you can use in other software.

2.3 Can I produce/extract 3D surfaces (meshes) out of CT/MRI data using MorphoDig ?

No. To extract 3D meshes CT/MRI data sets, you have to use another software. However, you can edit 3D Meshes in various ways using MorphoDig.

2.4 Is there a CTRL-Z functionality ?

Yes, definitely. Most actions can be undone and redone.

Chapter 3

Main window, Open data, Save data, Undo-Redo actions

Contents

3.1 Main window.

MorphoDig's main window is organized as shown in Fig. 3.1 p.14. The most commonly used controls are directly accessible nearby the main 3D rendering window, and are sorted by function.

3.2 Open and save data

- There are three main ways to open data in MorphoDig. First, you can open specific files via the menu "File". You may also open data by clicking on the "open" () button inside the main window (or press "CTRL +O"). You may also drag and drop files within the main 3D window. STL, PLY and VTK polydata surface files can be opened by MorphoDig, and many files specific to that software (files, curves, positions, color maps, tag maps, orientation-helper labels etc.).

- In most cases, only selected objects (or some of their properties) can be saved; most "save" functionalities will not affect unselected objects. There are two main ways to save data in MorphoDig. First, you can save specific files through the menu "File". You may

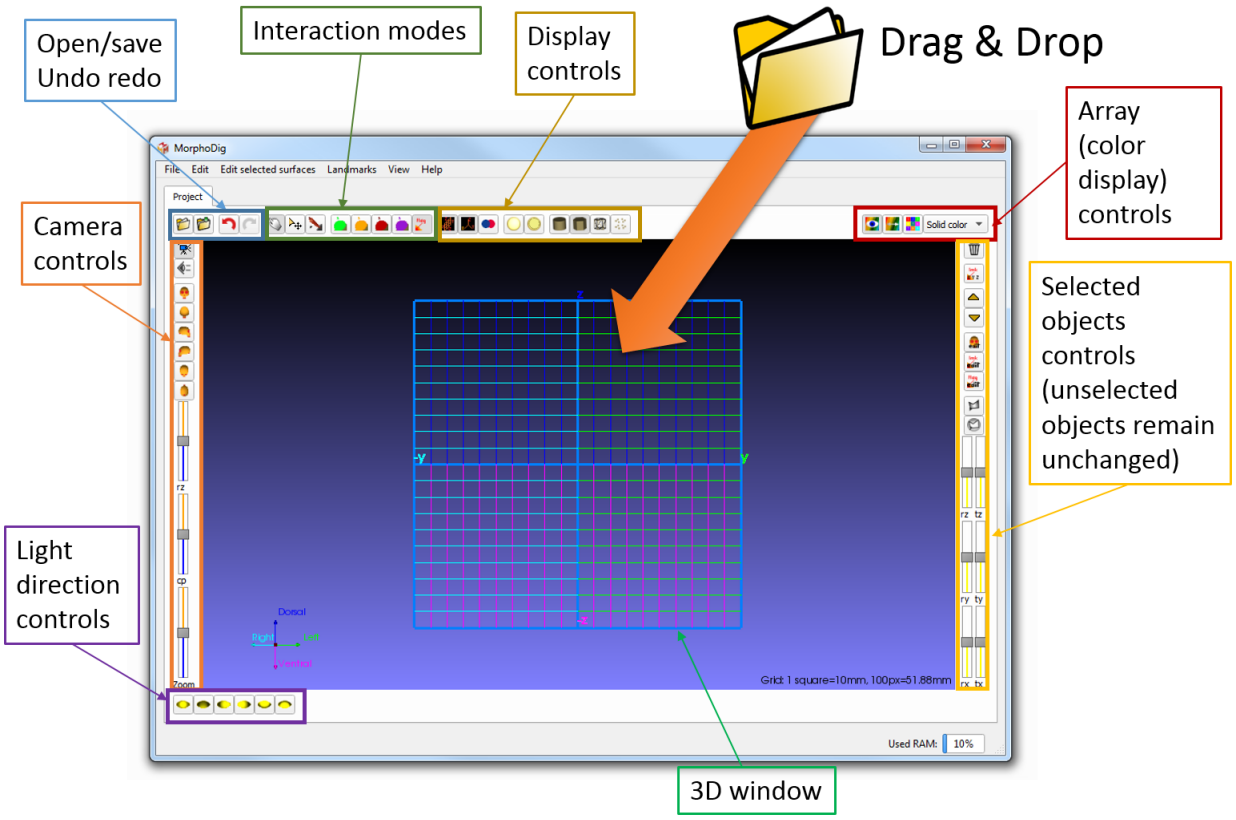


Figure 3.1: MorphoDig's GUI. Controls are organized around the main 3D rendering window.

also save a project with the button "save project" () inside the main window. Only selected objects will be considered when saving a project.

3.3 Undo-Redo actions

- Most actions, such as opening data, deleting data, selecting/unselecting objects, placing landmarks, modify the position of landmarks/surfaces, surface tagging etc. can be undone by clicking the "undo" button ().
- The same actions can be redone by clicking on the "redo" button ().

Chapter 4

Interactions and color display.

Contents

4.1 Interacting with objects

One very important aspect of MorphoDig's design is that most interactions or modifications of opened objects can only be done when these objects are selected. Selected objects (3D surfaces and landmarks) are always drawn in "grey". All currently opened objects can be selected by pressing CTRL+A. All currently opened objects can be unselected by pressing CTRL+D. Objects can also be selected/unselected using the right mouse button, depending on the currently active interaction mode (see below).

4.2 Interactions modes

See Fig. 3.1 p.14 to find the location of the "Interaction modes" controls in the main Window.

4.2.1 Camera mode

 “Camera mode" is the default interaction mode, and is active on startup. When active, left and middle mouse button drags result in camera rotation/translation, respectively. Right mouse button drag results in object selection/unselection. Pressing "ESC" switches between object mode and camera mode.

4.2.2 Object mode

 When active, left and middle mouse button drags result in object rotation/translation, respectively. Right mouse button drag results in object selection/unselection. Pressing "ESC" switches between object mode and camera mode.

4.2.3 Landmark mode

 When active, only landmarks can be selected/unselected via right mouse button drag. This mode is useful when editing/placing landmarks. Left and middle mouse button drags result in camera rotation/translation, respectively.

4.3 Landmark setting modes

Landmarks can be set on surfaces by pressing “L" + left mouse click. Four series of landmarks can be set with MorphoDig: “normal" landmarks, “target" landmarks, “curve node" landmarks and “curve handle" landmarks. Additionally a fourth special landmark series (“flag" landmarks) can be used to label surface structures.

4.3.1 Normal landmark mode

Press “  " to activate this mode (this mode is active by default)

4.3.2 Target landmark mode

Press “  " to activate this mode

4.3.3 Curve node mode

Press “  " to activate this mode

4.3.4 Curve handle mode

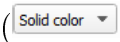
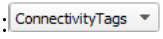
Press “  " to activate this mode

4.3.5 Flag landmark mode

Press “  " to activate this mode

4.4 Unselected surfaces color display

See Fig. 3.1 p.14 to find the location of the "Array controls", which impact the color display of the surfaces. A given unselected surface can be colored:

- using a uniform “solid" color (Fig. 4.1-A p.20). Scalar display mode must be deactivated or active array combo box set to “Solid color" ()
- according to the scalar values (= numbers) associated at the vertices (Fig. 4.1-B p.20). Array display mode button must be pressed (), and a Scalar array must be selected (ex: ). The way scalar arrays are translated into colors can be set up using color "Lookup tables" (LUT), also referred to as color transfer functions. The "Scalar rendering options" window can be opened by clicking on "  ".
- according to the tag values (= integers) associated to the vertices (Fig. 4.1-C p.20). Array display mode button must be pressed (), and a Tag array must be selected (ex: ). The way Tag arrays are translated into colors can be set up using color "Lookup tables" (LUT), also referred to as color transfer functions. The "Tag options" window, in which lookup tables associated to tag arrays can be edited, can be opened by clicking on "  "
- according to RGB values directly associated to the vertices (Fig. 4.1-D p.20). Array display mode button must be pressed (), and a RGB array must be selected (ex: )

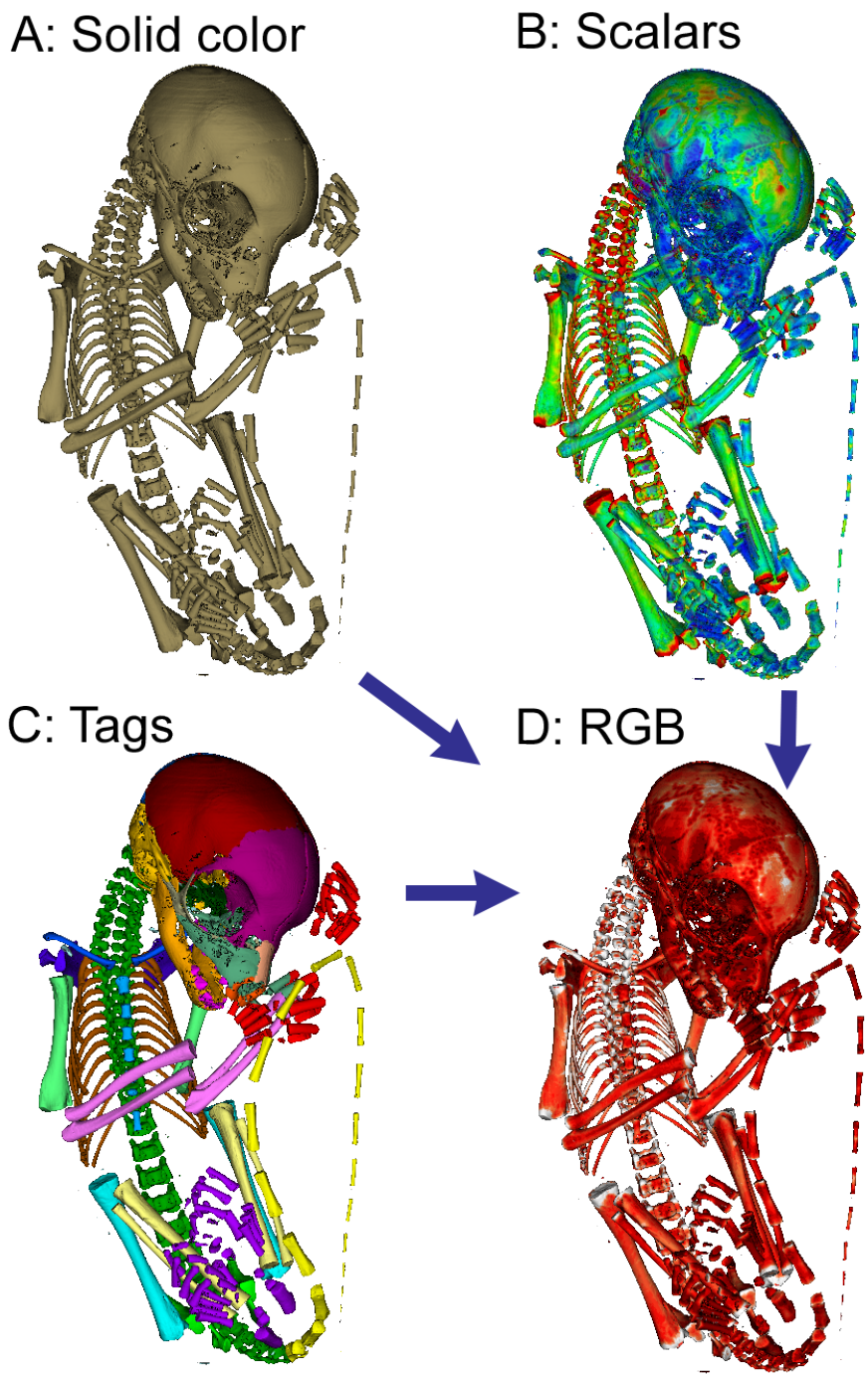


Figure 4.1: Color modes for a given surface. Single surface representation of the skeleton of a newborn *Lemur catta*. A: "Solid color mode" mode : a uniform color is used to represent the whole surface. B: "Scalar array" representation mode. Numbers (here bone thickness) are associated to each vertex of the surface, and are rendered using a color map. C: "Tag array". Integers are associated to each vertex (here different integer values are associated to different groups of bones), and a color map is used to colorize the specimen. D : "RGB array" : red, green, blue and transparency information is directly associated to each vertex: no color map is used in that case. In MorphoDig, RGB arrays can be produced out of solid color information, scalar arrays and tag arrays. PLY file produced via surface scans often contain surch RGB information.

Chapter 5

Keyboard and mouse controls

Contents

5.1 Keyboard and mouse controls

Ctrl+A	Selects all objects
Ctrl+D	Unselects all objects
Ctrl+Z	Undo last action
Ctrl+Y	Redo last action
L+ left click	Creates a landmark (either “normal", “target", “curve node", “curve handle" or “flag" landmark).
L + right click	If a single landmark or flag is selected, its position is moved. Nothing happens if no landmark nor flag is selected or if more than one landmark or flag are selected
Left mouse button drag	Camera mode : camera rotation. Object mode : object rotation. Landmark mode : camera rotation.
Ctrl + left mouse button drag	Camera mode : object rotation. Object mode : camera rotation. Landmark mode : object rotation.
Right mouse button drag	Draws a yellow rectangle. Once right button is released, all objects (surfaces and landmarks) falling inside the rectangle get selected/unselected, depending on their initial selection status
Middle mouse button roll (roll wheel)	Zoom / unzoom
Middle mouse button drag	Camera mode : camera translation Object mode : object translation Landmark mode : camera translation
Ctrl + middle mouse button drag	Camera mode : object translation Object mode : camera translation Landmark mode : object translation
« Del »	All selected objects are deleted
T + left click	Picks a vertex and tags corresponding surface with active tag value (a tag array must be active, and tag mode must be activated)
T + right click	Picks a vertex and tags corresponding surface with active tag value (a tag array must be active, and tag mode must be activated). The only potentially affected vertices are those having initially the same color as that of the picked vertex.

5.2 Additional controls

Additional controls are available when using “lasso cut" or “lasso tag" (lasso mode should be active):

Left mouse click and drag	Draws the lasso polygon contour
Left mouse release	all the vertices falling within the polygonal region (outside or inside the polygon) are either: - given the color corresponding the active tag (lasso tag) - inserted into a new object (lasso cut inside). - deleted from the new object (lasso cut outside). See lasso cut (section 6.4.7 p.36) and lasso tag (section 12.5.3 p.129) sections for further information.

Chapter 6

Main window controls

Contents

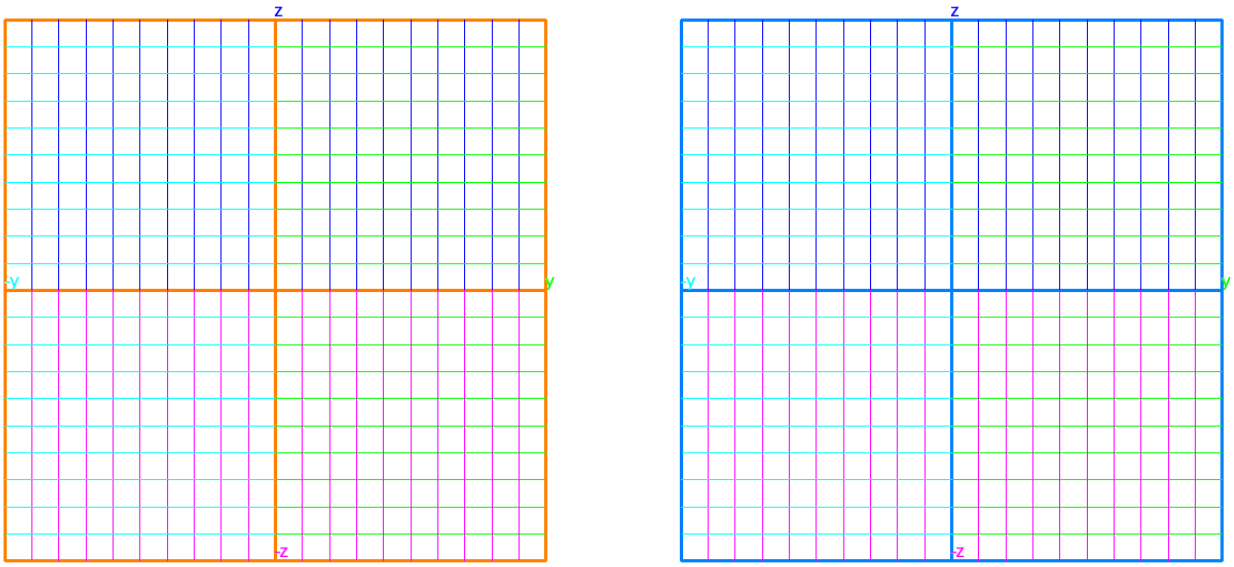


Figure 6.1: Grid display color. Left: when the camera revolves around the origin of the coordinate system (x=0, y=0, z=0), the grid outline is displayed in orange. Right: when the camera revolves around the center of mass of all opened objects, the grid has a blue outline.

See Fig. 3.1 p.14 to have an overview of the controls described in this chapter.

6.1 Camera controls

Most camera related controls lay on the left part of the main window. See Fig. 3.1 p.14 to find the location of the "Camera" controls in the main Window.

6.1.1 Camera rotation center

By default, the camera rotates around the origin of the coordinate system (x=0, y=0, z=0), but by pressing "" / "", the camera will revolve around the center of mass of all opened objects. The latter option is useful when the centre of mass of an object (or of several ones) is far from the origin of the coordinate system. The grid is drawn using different colors depending on the camera rotation centre (see Fig. 6.1 p.26).

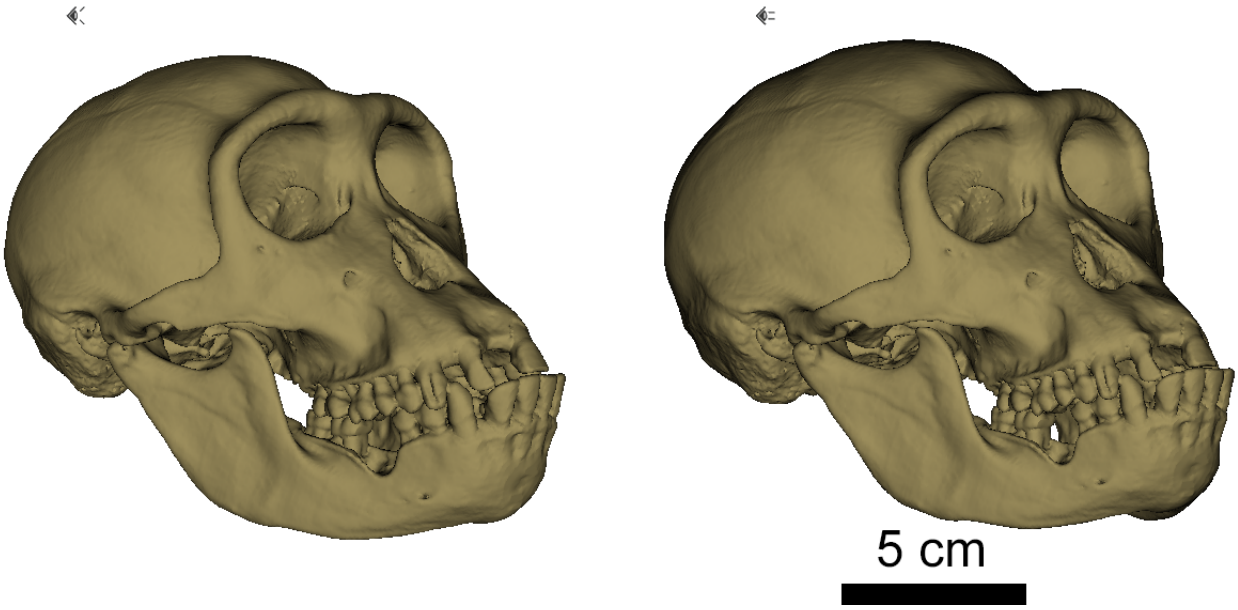


Figure 6.2: Perspective and Orthographic camera projection modes. Skull of *Pan panis-cus*. Left: perspective camera projection mode: closer structures appear larger than farther structures. Right: orthographic camera projection mode: a uniform relation exists between pixel size and real world dimensions, making it possible to place a scale bar.

6.1.2 Orthographic or perspective camera projection

You can switch between orthographic and perspective projection mode by pressing the "  " or "  " toggle button (see also see Fig. 6.2 p.27). Note that in orthographic camera projection mode  , when the grid is displayed, the relationship between pixel size on the screen and real world dimensions appears on the right bottom corner of the screen: 

.

6.1.3 Camera orientation

6 camera positions are predefined :

 view object from right side

 view object from left side

 view object from front side (default camera position)

 view object from back side

 view object from above

 view object from below

6.1.4 Camera rotation around “z" viewing axis

To do so, you may use the slider lying around the center of the left panel of the main window.

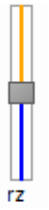


Figure 6.3: Camera “z" rotation slider

6.1.5 Clipping plane slider

In some cases, you may need to displace the viewing clipping plane. To do so, use the slider lying centrally in the left panel of the main window. See 6.2.4 p.29 for additional clipping plane functionalities.

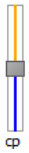


Figure 6.4: Camera clipping plane slider

6.1.6 Zoom

There are three main ways to modify the “zoom" in MorphoDig :

- You may use the zoom slider laying in the lower part of the left panel of the main window.
- You may set manually the display scale (Edit → Edit size unit and grid spacing, then define the display scale: 100 pixels in size unit). This option is only available in orthographic projection mode .
- You may use the middle click mouse roll button (roll the wheel).

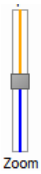


Figure 6.5: Zoom slider

6.2 Display controls

The display controls mentioned in this section are situated on the top part of the main window (see Fig. 3.1 p.14).

6.2.1 Grid

Press  to show / hide the grid. Default grid size is 1 cm / square. Grid size can be edited manually (viewing opt. → Grid size). Switching between the 6 camera predefined positions defined above ( ,  ,  ,  ,  and ) will affect the plane in which the grid is drawn.

6.2.2 Coordinate system orientation helper

Press  to show / hide the coordinate system orientation helper lying on the bottom left corner of the main 3D window. By default, the labels are defined the following way:
+z axis : dorsal side
-z axis : ventral side
+y axis : left side
-y axis : right side
+x axis : anterior side
-x axis : posterior side.
You may edit these labels depending on your preferences (for instance, depending on the structure you are working with, you may need to set “+y" to “labial", and “-y" to “lateral"). To edit orientation labels, click on “Edit → Edit orientation labels."

Figure 6.6: Orientation helper

6.2.3 Anaglyph

Press  to activate/deactivate anaglyph 3D rendering. See Fig. 6.7 p.30. You then need to wear "Red/Blue"  glasses to visualize objects in 3D.

6.2.4 Clipping plane

The button  or  permits to adjust / readjust the position of the clipping plane at predefined positions (see for instance Fig. 6.8 p.30):

-  : the clipping plane is placed at z = 0 (all objects having a z coordinate along z viewing axis smaller than 0 are hidden).
-  : the clipping plane is replaced at its original value : z= - camera.far / 2. This value permits to view objects having positive and negative coordinates along z viewing axis.

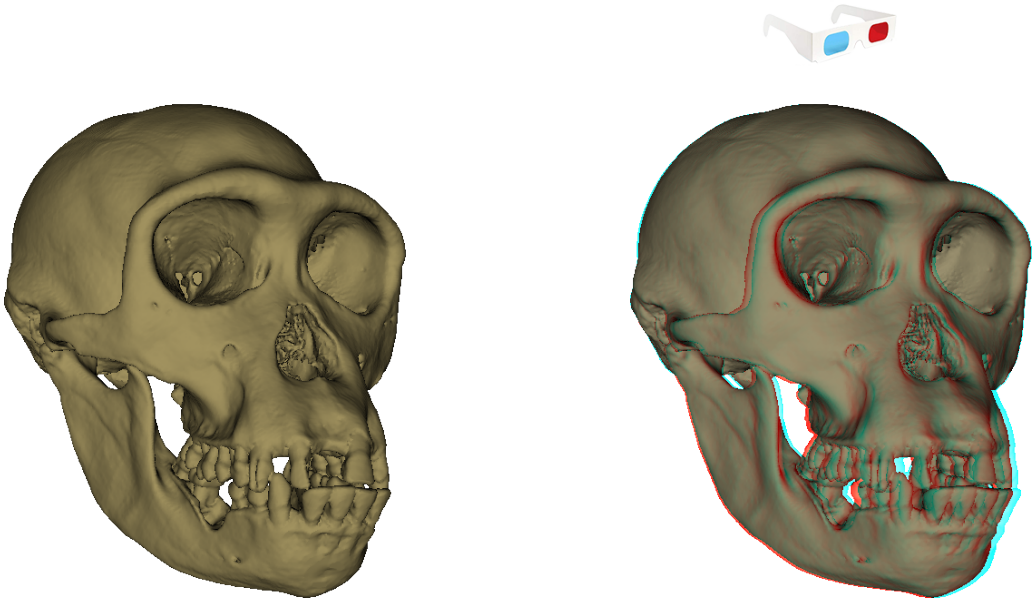


Figure 6.7: Anaglyph display mode. Left: normal display mode. Right: anaglyph display mode. Cranium of the holotype specimen of *Pan paniscus*.

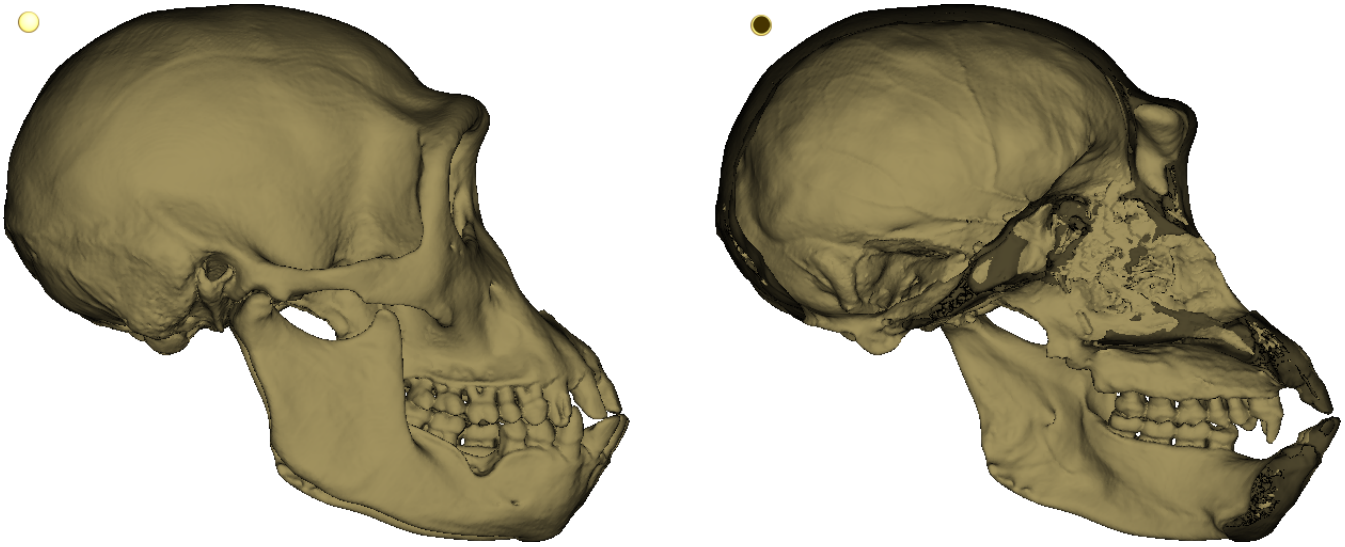


Figure 6.8: Clipping plane display mode. Cranium of the type specimen of *Pan paniscus*. Left: normal display mode. Right: clipping plane display mode "on" : permits to visualize inner structures.

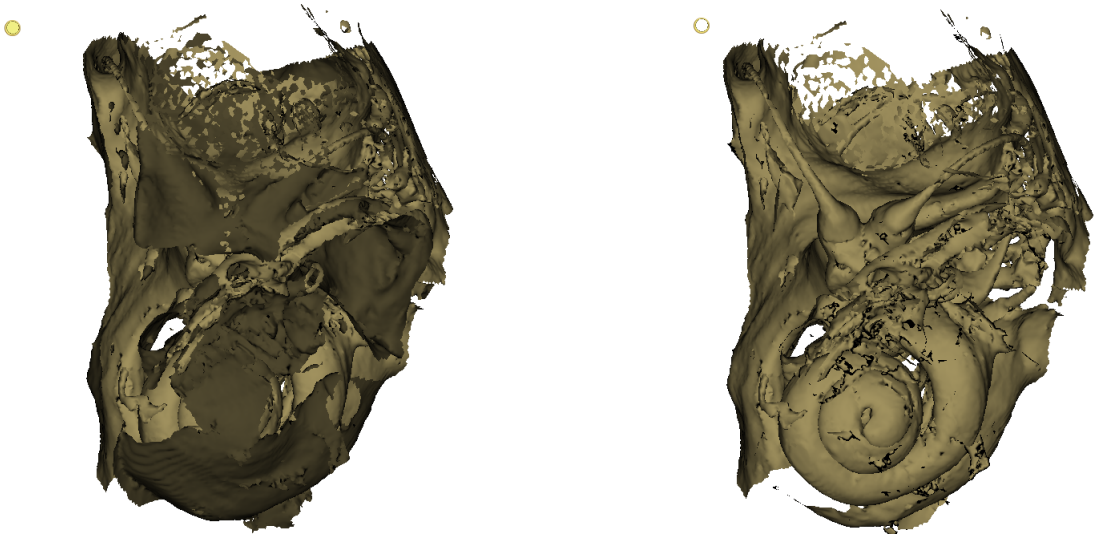


Figure 6.9: Backface culling display mode. Petrosal bone of *Eulemur mongoz*. Left: normal display mode: both frontfaces (lighter) and backfaces (darker) are displayed. Right: backface culling mode activated: only frontfaces are displayed, making it easier to visualize the bony labyrinth.

6.2.5 Backface culling

Hiding backfaces of a surface is useful in some cases to visualize inner structures of a biological object (see for instance Fig. 6.9 p.31). The button / permits to visualize/hide backfaces :

- : both sides of a given surface object are rendered.
- : backfaces are hidden (only frontfaces are visible).

6.2.6 Surface 3D representation controls

4 different options exist to draw 3D surfaces :

: point normals display mode (Gouraud shading), see Fig. 6.10-A.

: triangle normals display mode, see Fig. 6.10-B.

: wireframe representation display mode, see Fig. 6.10-C.

point cloud display mode, see Fig. 6.10-D

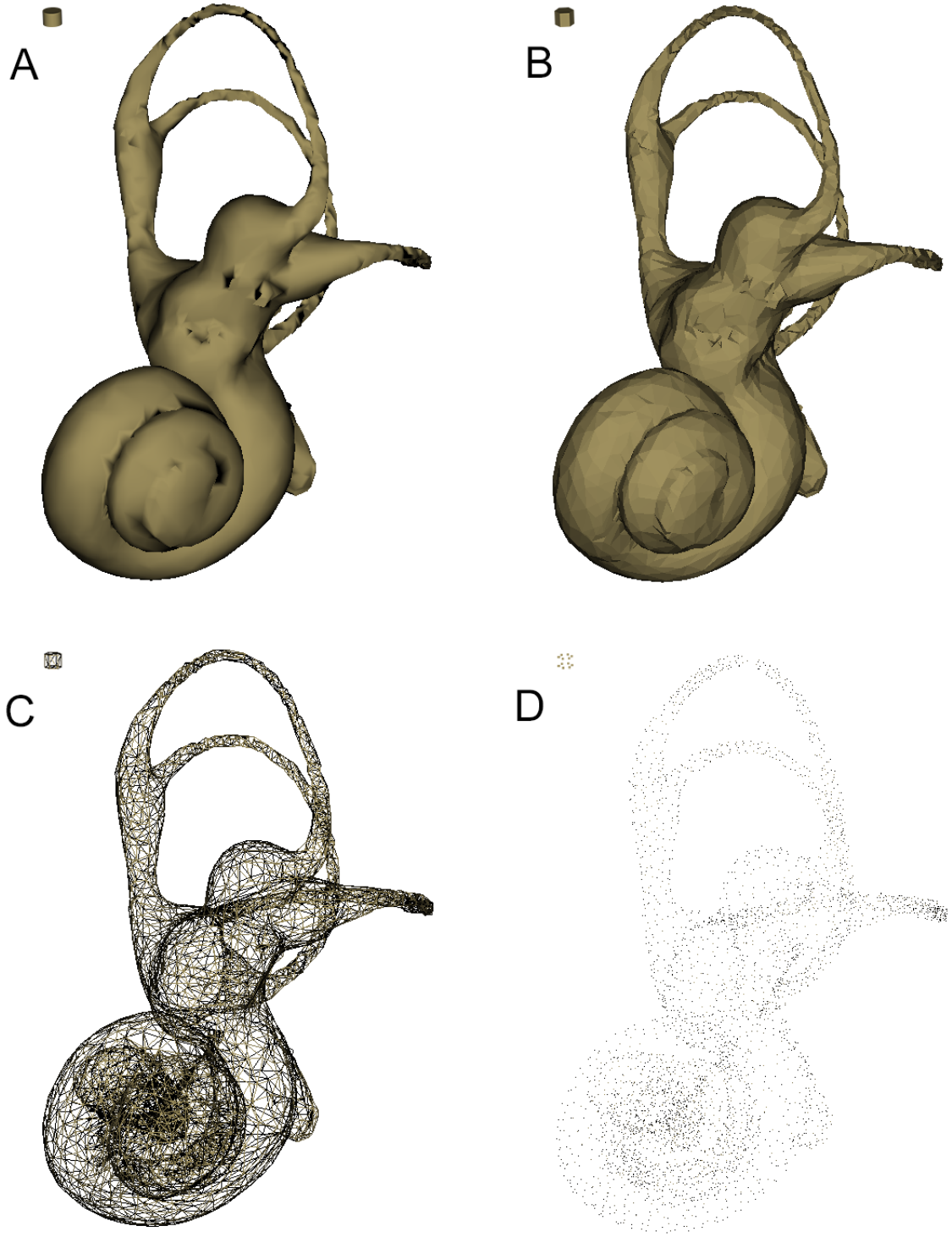


Figure 6.10: Display modes available for a given surface. Single surface representation of the left bony labyrinth of a *Lepilemur dorsalis*. A: "Point normales" mode (Gouraud shading). B: "Triangle normales" mode. C: "Wireframe" mode. D: "Point cloud" mode.

6.3 Light direction controls

See Fig. 3.1 p.14 to see where the "light controls" are situated within MorphoDig's main window.

6 lightning orientations are predefined :


light from right viewing side


light from left viewing side


light from front viewing side


light from back viewing side


light from above


light from below

6.4 Object controls

See Fig. 3.1 p.14to see where the "object controls" are situated within MorphoDig's main window. Remind that only selected objects are affected by these controls.

6.4.1 Delete selected objects

Clicking on "  " deletes all selected objects. This action can be undone/redone.

6.4.2 Create landmark at X,Y,Z

Clicking on "  " opens the "Create landmark" window (Fig. 6.11 p.34), in which a new landmark can be created a specified coordinates. This action can be undone/redone..

6.4.3 Decrease / Increase landmark number

Clicking on  will increase (= move down in list, if possible) the landmark number of all selected landmarks. Clicking on will decrease (=move up in list, if possible) the landmark number of all selected landmarks. These buttons make it possible to reorder landmarks.

6.4.4 Edit first selected surface

Clicking on "  " opens the "Edit first selected surface" window (Fig. 6.12 p.35), in which several properties of a given surface object can be edited.

- Object name : modifies the name of the object.
- Next object and preceding object  : saves currently settings and fetches next/preceding surface.

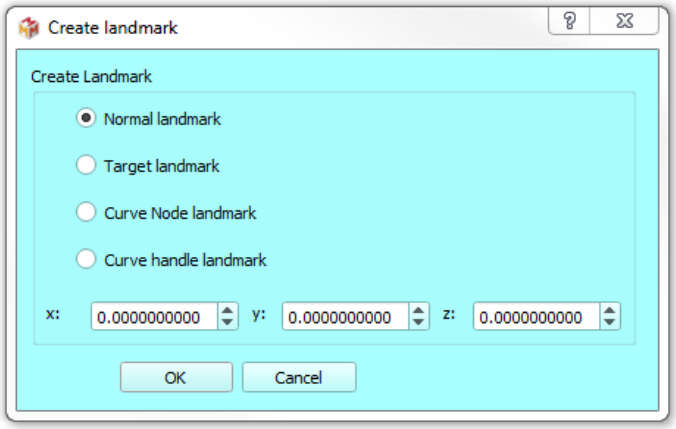


Figure 6.11: Create landmark at a given X,Y,Z coordinate.

- Object solid color : modifies the "solid color" property of this object.
- Object translucency : modifies the translucency this object. - Object matrix : current position/rotation matrix of the object.
- Reinit matrix: set current matrix to identity.
- Apply matrix to all selected surfaces : set position/rotation matrix to all other selected surfaces.
- Existing arrays : list of all scalar/RGB color/tags arrays associated to the vertices of the surfaces.
- Edit array name : modifies the name of currently selected array
- Duplicate array : duplicates currently selected array
- Delete array : deletes currently selected array
- Ok : save settings and closes window

6.4.5 Edit first selected landmark

Clicking on "lmk edit" opens the "Edit first selected landmark" window (Fig. 6.13 p.35), in which landmark coordinates of landmarks can be manually edited. The center of rotation of the camera can be set to the x,y,z coordinates of a given landmark by pressing on "".

6.4.6 Edit first selected flag

Clicking on "Flag edit" opens the "Edit first selected flag" window (Fig. 6.14 p.36), in which several properties of a given flag can be manually edited.

- Flag name : modifies the label of this flag.

- X, Y, Z: 3D coordinates of the flag.
- Next flag and preceding flag : saves currently settings and fetches next/preceding flag

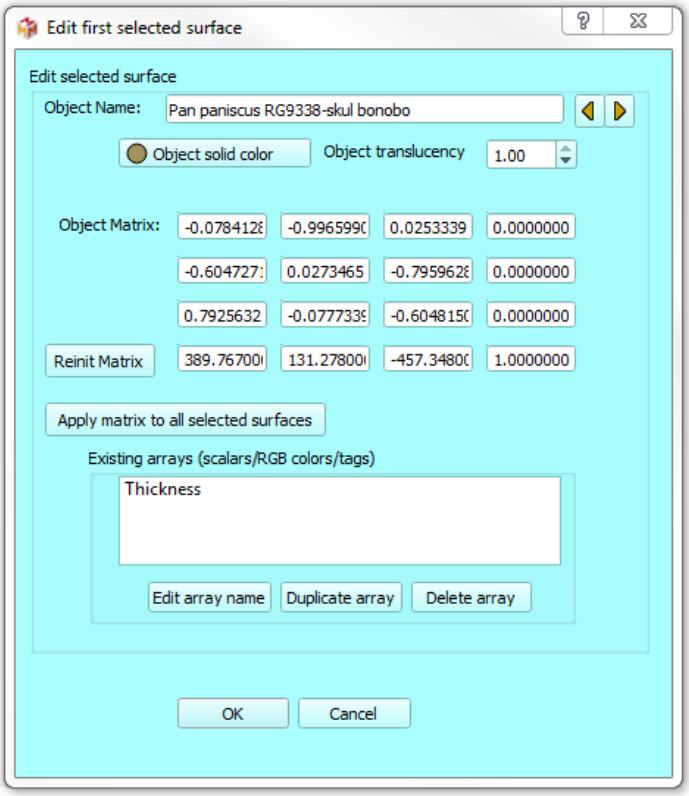


Figure 6.12: Edit first selected surface window.

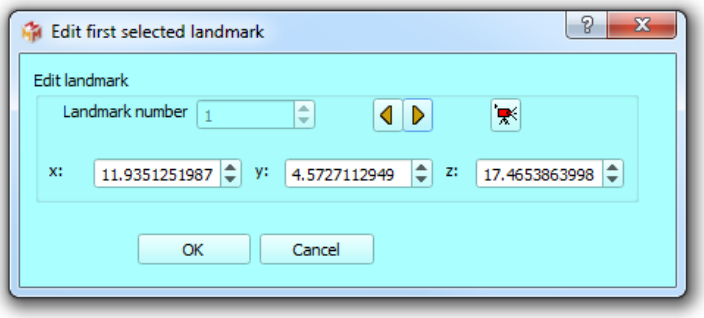


Figure 6.13: Edit first selected landmark window.

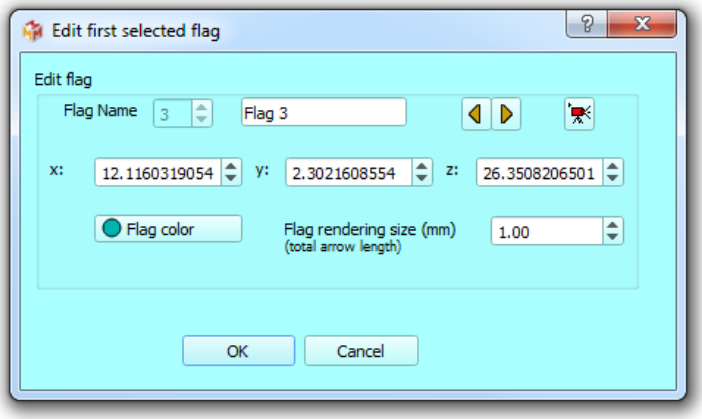


Figure 6.14: Edit first selected flag window.

- Flag color : modifies the "color" of the currently selected flag.
- Flag rendering size : modifies the length of the flag.

The center of rotation of the camera can be set to the x,y,z coordinates of a given flag by pressing on "".

6.4.7 Lasso cut selected surfaces

You may cut through an input selected surface using this option (see Fig. 6.15 p.37). Once "" (lasso cut keep inside) or "" (lasso cut keep outside) has been pressed, the mouse cursor changes and you can start drawing the lasso section by dragging the mouse while the left button is pressed. When the left button is released, the lasso cut operation starts: a new object is created, which contains all the parts of the selected object laying inside or outside the drawn polygon, respectively.

6.4.8 object rotation and translation sliders

As seen earlier, selected objects can be translated and rotated using the mouse left and middle buttons (in landmark and camera selection modes, you also need to maintain “CTRL" button pressed while dragging the mouse to achieve rotation and translation of selected objects). Alternatively, you may also use the following controls to accomplish rotation and translation of selected objects. Rotation is performed around the global center of mass of all currently selected objects.

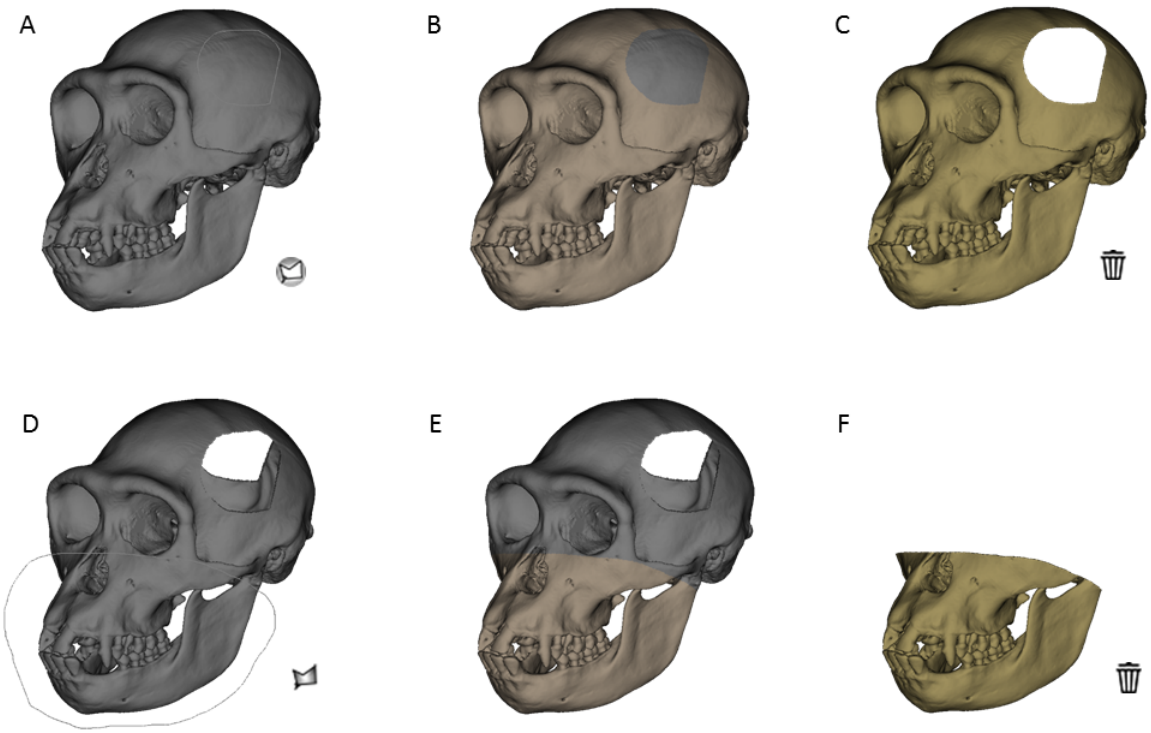


Figure 6.15: A: "Lasso keep outside" button is pressed, then a polygon is drawn over the selected object (grey) by maintaining left mouse button pressed and dragging the mouse. B: Left mouse button has been released : a new surface object is created. C: "Delete" has been pressed : only the newly created object remains. C: "Lasso keep inside" is pressed, then a polygon is drawn over the selected object (grey) by maintaining left mouse button pressed and dragging the mouse. E: Left mouse button has been released : a new surface object is created. F: "Delete" has been pressed : only the newly created object remains.

Rotation around and translation along “Z" viewing axis

These controls are extremely useful, as there is no way to achieve rotation round « z » viewing axis or translation along “z" viewing axis using the mouse.
To do so, use the tz and rz sliders (see Fig. 6.16 p.39).

Rotation/translation around/along “Y" viewing axis

To do so, use the ty and ry sliders (see Fig. 6.17 p.40).

Rotation/translation around/along “X" viewing axis

To do so, use the ty and ry sliders (see Fig. 6.18 p.41).

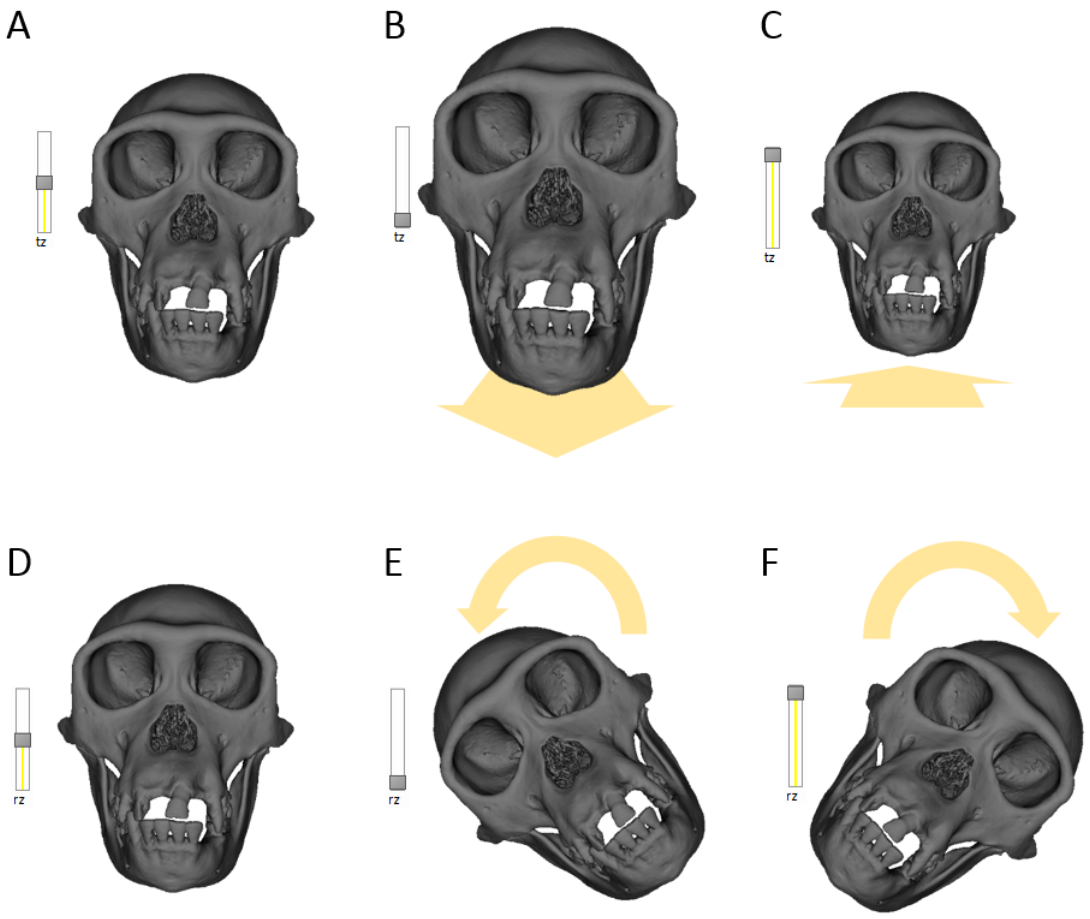


Figure 6.16: A: object and tz slider are in initial position. B: tz slider is moved down: the selected object moves forward. C: tz slider is moved up. The selected object moves backward. D: object and rz slider are in initial position. E: rz slider is moved down: the selected object rotates around the "roll" axis counter-clockwise. F: rz slider is moved up. The selected object rotates around the "roll" axis clockwise.

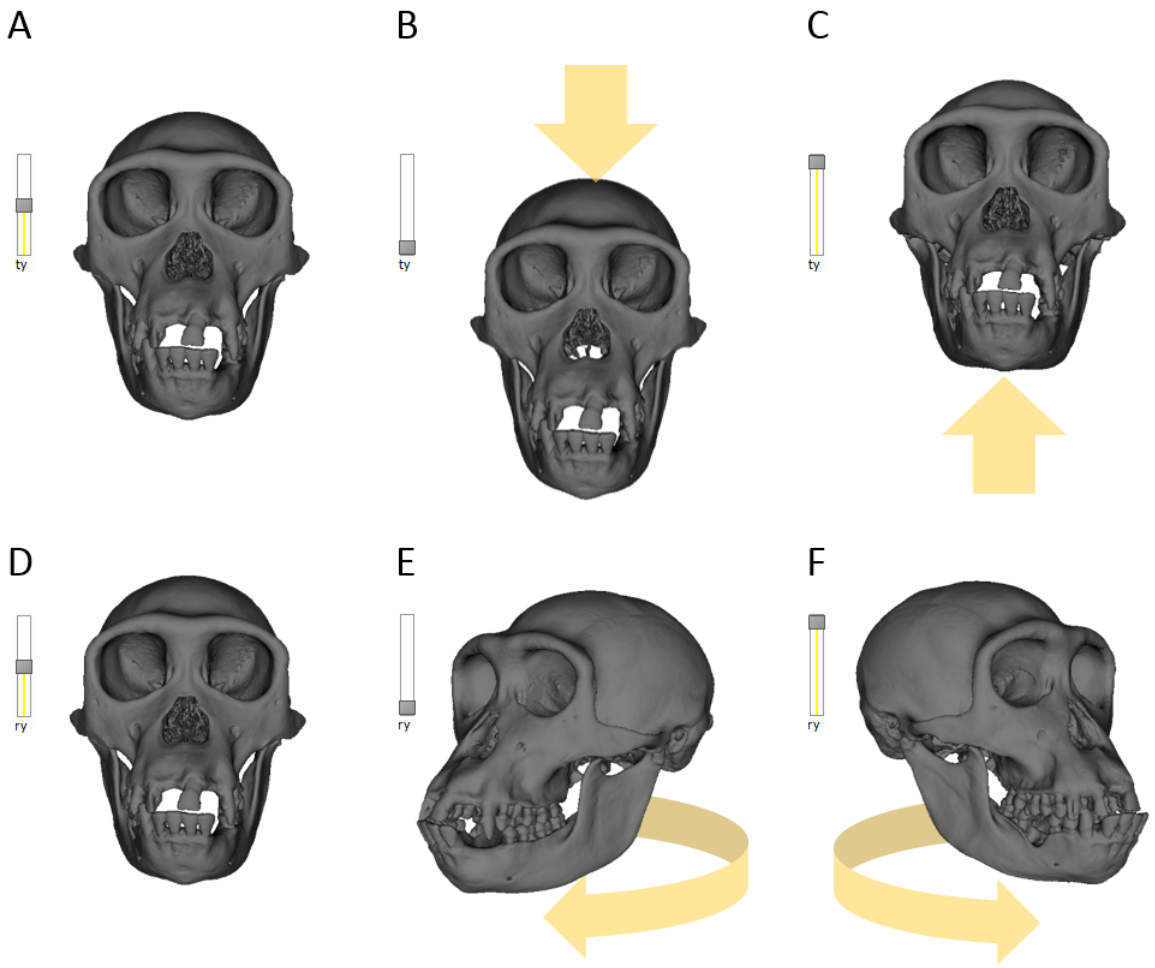


Figure 6.17: A: object and ty slider are in initial position. B: ty slider is moved down: the selected object moves downward. C: ty slider is moved up. The selected object moves upward. D: object and ry slider are in initial position. E: ry slider is moved down: the selected object rotates around the "yaw" axis clockwise. F: ry slider is moved up. The selected object rotates around the "yaw" axis counter-clockwise.

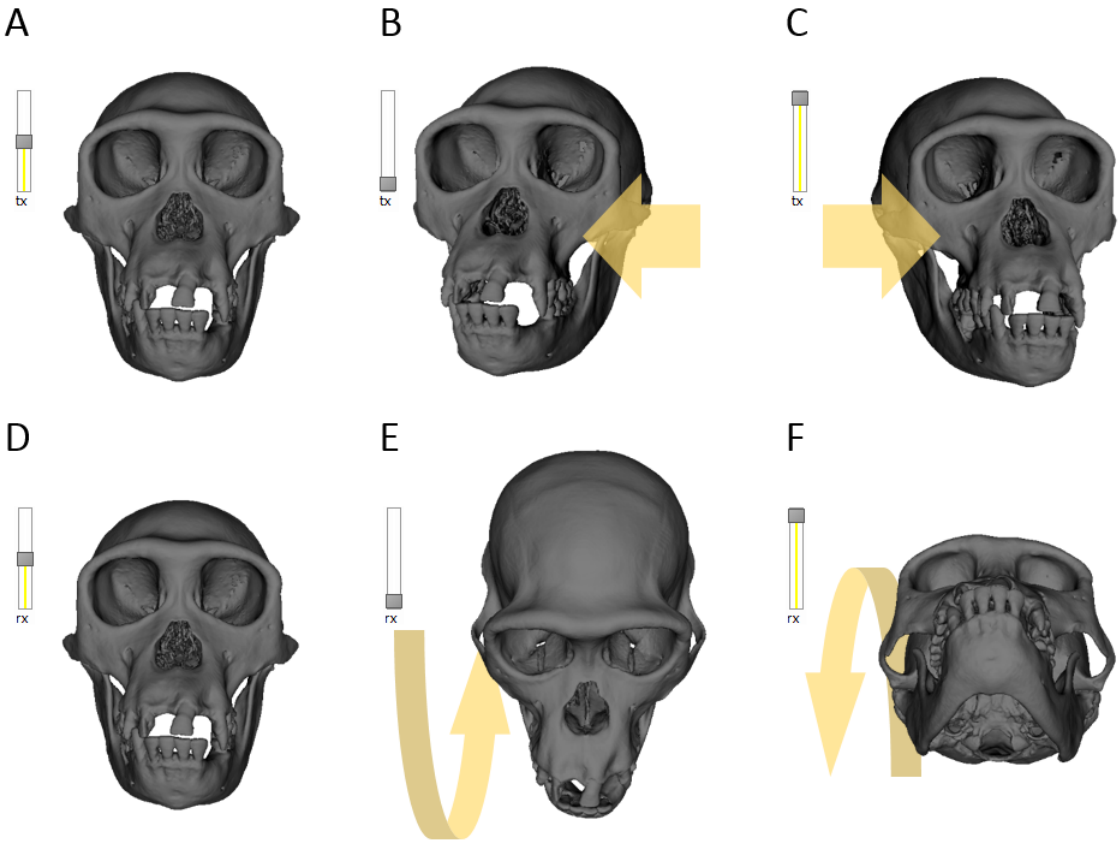


Figure 6.18: A: object and tx slider are in initial position. B: tx slider is moved down: the selected object moves leftward. C: tx slider is moved up. The selected object moves rightward. D: object and rx slider are in initial position. E: rx slider is moved down: the selected object rotates around the "pitch" axis clockwise. F: rx slider is moved up. The selected object rotates around the "pitch" axis counter-clockwise.

Chapter 7

Menu File

Contents

```
exploded.ori
exploded.tag
exploded.flg
exploded.stv
Hippopotamus_amphibius_braincase.vtk
Hippopotamus_amphibius_braincase_exploded.pos
0.250980 0.482353 0.494118 1.000000
Hippopotamus_amphibius_bulla.vtk
Hippopotamus_amphibius_bulla_exploded.pos
0.647059 0.556863 0.086275 1.000000
Hippopotamus_amphibius_petrosal.vtk
Hippopotamus_amphibius_petrosal_exploded.pos
0.764706 0.356863 0.000000 1.000000
Hippopotamus_amphibius_sinus.vtk
Hippopotamus_amphibius_sinus_exploded.pos
0.470588 0.200000 0.568627 1.000000
```

Figure 7.1: Example of project .ntw file containing references to orientation labels, tags labels and colors, flags and landmarks files.

7.1 Project

When working with multiple surface objects, loading surfaces and associated positions one by one becomes fastidious. Besides, after having digitized landmarks, flags, after having set tag names and colors, or after having defined orientation labels, you may wish to open or save this information along with surface files. You may open and save series of surface files and associated position matrices, landmarks, flags, tag colors and labels and orientation labels using this menu. “Project" files (.ntw) files are organized the following way (see Fig. 7.1 p.45):
- Optional: name of orientation label file (.ori)
- Optional: name of tag file (.tag)
- Optional: name of flag file (.flg)
- Optional: name of landmak file (.lmk, .ver, .stv or .cur)
- Name of surface 1 file
- Name of position 1 file associated to surface 1
- Surface 1 RGB color and transparency
- Name of surface 2 file
- Name of position 2 file associated to surface 2
- Surface 2 RGB color and transparency (etc...)

Surface files can be of the following types : .stl, .vtk and .ply
“.ntw" files can be constructed manually, providing that the referred surface and position files exist.

7.1.1 Open project

Loads a .ntw file

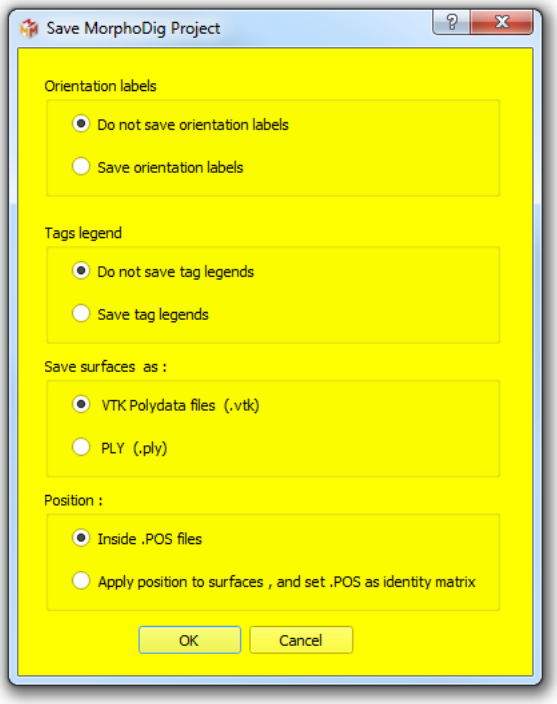


Figure 7.2: Save project window.

7.1.2 Save project

Behavior: only selected elements (surface objects, landmarks, flags...) are saved within projects. If at least one opened element has been selected, the Save project window (Fig. 7.2 p.46) opens and asks whether :

- orientation labels should be saved along with the project (by default, orientation labels are not saved, as orientation labels are quite rarely set by users).
- tag legends and colors (tag maps) should by saved. By default, .tag file are not saved along with the project, as tag setting is not a common task.
- save surfaces as. VTK polydata or PLY. We advise you to save your surfaces as VTK polydata. In the contrary cases scalar, tag arrays and all RGB arrays having a name different from "RGB" will be lost.
- save position inside .POS file, or apply the position to surfaces. We advise you to save position inside .POS file (otherwise, the initial orientations of your surfaces will be lost).

Each surface file will be given the name of the original file. Each position file will be given a name which starts with the name of the associated surface and ends with the name of the project (1 exception: when the project contains only one surface and the project's name is the same as that of the surface. In that case, .pos file names are not postfixed with the name of the project). In the .ntw file example shown in Fig. 7.1 p.45, the surface files are

- “Hippopotamus_amphibius_braincase.vtk"
- “Hippopotamus_amphibius_bulla.vtk"
- “Hippopotamus_amphibius_petrosal.vtk"
- “Hippopotamus_amphibius_sinus.vtk"

and the project name is “exploded.ntw". The advantage of naming position files that way is you may construct different .ntw files with different associated surface files using a same set of surfaces. Requirement : all selected surfaces saved via this option need to have distinct names. Note : When working with “project" files, if several surface objects have the same name, you may need at some point to rename some of the object surfaces in order to make sure that all surface objects have a distinct name. To do so, select one surface, click on  : the “Edit first selected surface" window appears (See Fig. 6.12 p.35 in preceding chapter).

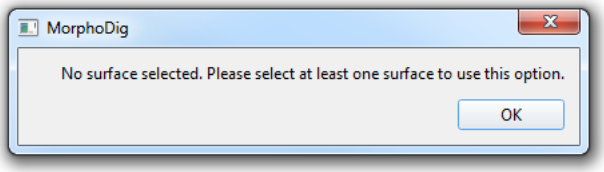
7.2 Surface

7.2.1 Open surface

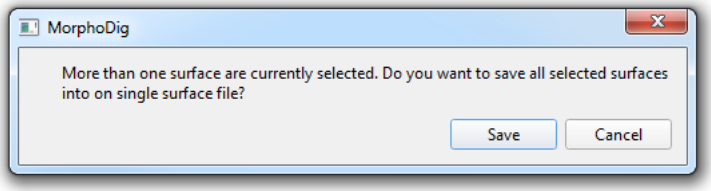
STL, PLY and VTK polydata surfaces can be open via this menu. MorphoDig does not manage textures associated with surface files. When opening a PLY file containing RGB colors (for instance a file painted manually or automatically with “MeshLab", or some laser scanner surface files) or a VTK file containing RGB arrays, these colors are placed inside a (or several) "RGB" array(s).

7.2.2 Save surface

Selected surfaces can be saved into files. If no surface is selected, the following message appears:



If at least 2 surfaces are selected, the following message shows up:



Save selected surfaces in one single .PLY file

Options:

- File type: you can save .ply data in binary (little or big endian) or ASCII formats.
- Position : you can keep object original coordinate system or save the surface in its current position.
- Normals : you can choose whether you wish to save normals.
- Colors : RGB information associated to each vertex can be saved. You may decide either to keep current RGB color array (if that array exists). Alternatively, you may decide not to save the current RGB color array (if it exists). Finally, you may prefer to save/replace the current RGB color array as the current display color.

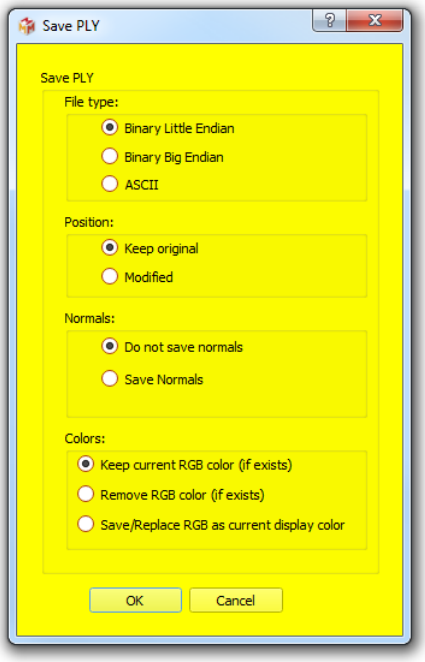


Figure 7.3: PLY save options window

Note that a "RGB" array (object rendering color, depending on which rendering mode you are using), depending on the chosen options, can be saved inside the .ply file. This means that Tag / Scalar / Object solid color can be exported and viewed in other software such as MeshLab.

Save selected surfaces in one single .VTK PolyData (.VTK or .VTP) file

VTK polydata surface file format is by far not as widespread as STL or PLY formats. However, it is extremely useful as it allows to store several scalar, tag and RGB arrays. Options:

- File type: you can save .VTK data in binary (little endian) or ASCII formats.
- Position : you can keep object original coordinate system or save the surface in its current position.
- Arrays: you decided which arrays (scalars, tags, RGB colors) should be saved inside the VTK polydata file.

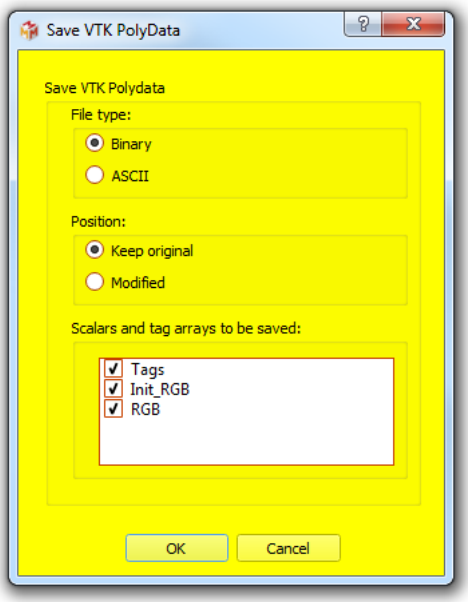


Figure 7.4: VTK save options window

Save selected surfaces in one single .STL file

Options:

- File type: you can save .stl data in binary (little endian) or ASCII formats.
- Position : you can keep object original coordinate system or save the surface in its current position

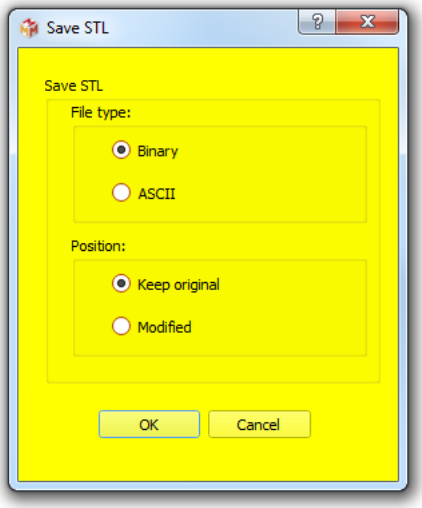


Figure 7.5: STL save options window

```
1.000000 0.000000 0.000000 0.000000
0.000000 1.000000 0.000000 0.000000
0.000000 0.000000 1.000000 0.000000
0.000000 0.000000 0.000000 1.000000
-0.035449 -0.919797 -0.390790 0.000000
0.079621 -0.392392 0.916346 0.000000
-0.996195 0.001369 0.087145 0.000000
10.772983 9.308582 -6.147681 1.000000
```

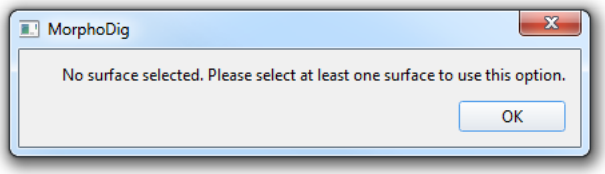
Figure 7.6: Example of .pos position file. The first 4 lines correspond to the aspect matrix, and the 4 last lines to the position matrix.

7.3 Position

In MorphoDig, surface position consists in two 4*4 square matrices: the first matrix is currently not read nor used anymore by MorphoDig, but is kept for compatibility issues with older versions of MorphoDig and of ISE-MeshTools (MorphoDig is a major redesign of an older software, ISE-MeshTools). This first matrix will disappear in a near future. The second matrix is the one that matters: the position matrix. It contains the rotation and translation information of 3D objects (3D surfaces and landmarks). These 2 matrices can be opened and saved in ".pos" format (see Fig. 7.6 p.51).

7.3.1 Open position for selected surfaces

If no surface is selected, the following message appears:



If at least 2 surfaces objects are selected, the same position will be given to them.

7.3.2 Open transposed position for selected surfaces

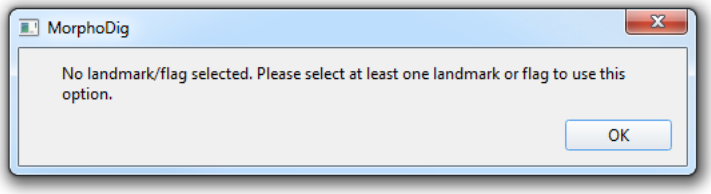
This option may be useful in the following case:

- Let us suppose that you did modify the position of a given surface and saved its position.
- Then you have saved the surface in its current modified position (= the original position of the surface is lost).

It may happen that you need to open the surface in its original position. To do so, you may apply this option (apply transposed position matrix to the modified surface).

7.3.3 Open position for selected landmarks/flags

If no landmark/flag is selected, the following message appears:



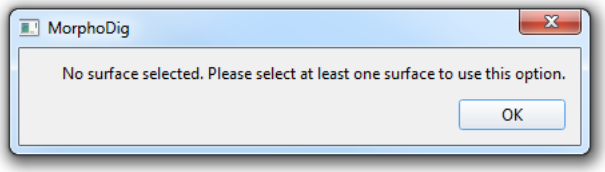
This options may be useful when you have digitized landmarks/flags on a given surface in its original position and orientation. Let us suppose that in a second step, you need to modify the position/orientation of that surface object and that you save it in a .POS file. You may then apply this .POS file to your set of landmarks/flags.

7.3.4 Open transposed position for selected landmarks/flags

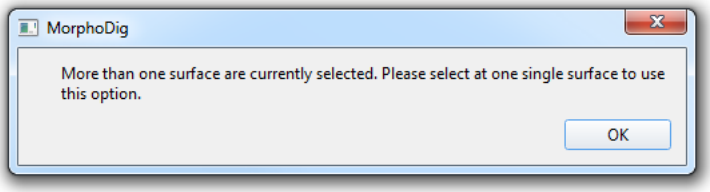
If you have digitized landmarks/flags on a surface with a modified position (let us assume that the position of this surface is saved within a .POS file), by using this option, you can replace this landmarks/flags dataset on the surface in its original orientation.

7.3.5 Save position for selected surfaces

Surface aspect and position matrices can be saved in ".pos" format. If no surface is selected, the following message appears:



If at least 2 surfaces selected, the following message shows up:



7.3.6 Edit manually position matrix

Clicking on "  " opens the "Edit first selected surface" window (Fig. 6.12 p.35), in which the position matrix of a given actor can be modified.

7.4 Landmarks

Landmarks can be set on surfaces by pressing “L" + left mouse click.

Two series of conventional landmarks can be set : “normal" and “target" landmarks. In the “normal" landmark mode (button  active), pressing “L" + left mouse click results in the creation a “normal" landmark (a light green one). In the “target" landmark mode (button  active), pressing “L" + left mouse click will create a “target" landmark (a yellow transparent one).

“Normal" and “target" landmarks can be loaded and saved.
Selected landmarks can be reordered using the following buttons. Pressing “  " will (try to) increase the number of all selected landmarks, while pressing ““"" will (try to) decrease their number, respectively.

MorphoDig can manage three types of landmark files: “.LMK", “.VER" and “.STV" files.

- .LMK files contain a series of lines, each line being constructed the following way (see Fig. 7.7 p.54): landmark name (without space or tab character), landmark coordinates. Note that each landmark name does not need to be of the form “landmark"+landmark number. Meanwhile, the name should not hold space or tab characters.

- .VER files contain a series of lines, each line being constructed the following way (see Fig. 7.8 p.54): landmark name (without space or tab character), landmark coordinates, landmark orientation.

- STV files may contain one or two series of line. The first line contains two integers, the first being the type of landmark (0 for “normal" or 1 for “target"), the second being the number of lines of landmarks of this type which are expected to follow, constructed the following way: landmark name (without space or tab character), landmark coordinates, landmark orientation. An example of .STV file containing both “normal" and “target" landmarks is given in Fig. 7.9 p.54. Note that the number of “normal" and “target" landmarks saved within a .STV file can differ.

```
landmark1: 6.179220 -1.516437 6.048982
landmark2: 5.260506 2.779373 6.386808
landmark3: 4.527496 4.214221 5.534292
```

Figure 7.7: Example of .LMK file

```
landmark1: 6.179220 -1.516437 6.048982 6.905368 -1.890056 6.626146
landmark2: 5.260506 2.779373 6.386808 5.862945 3.208343 7.059899
landmark3: 4.527496 4.214221 5.534292 5.015749 5.053222 5.774469
```

Figure 7.8: Example of .VER file

```
0 4
Landmark1: 2.90931 -0.299717 -0.185671 3.45482 -0.805245 0.482804
Landmark2: 2.67553 -0.284703 0.058657 3.258 -0.654435 0.782559
Landmark3: 1.74275 0.340697 -0.23797 1.37701 -0.547364 -0.516505
Landmark4: 1.22325 0.351425 -0.0563535 1.4208 -0.493512 0.440693
1 4
Landmark1: 2.97905 0.45206 -0.0969067 3.29039 0.883818 0.749646
Landmark2: 2.71838 0.412814 0.0735706 3.39374 0.301426 0.802597
Landmark3: 2.08129 0.570875 0.0844996 2.17525 0.169605 0.995627
Landmark4: 1.71463 0.436108 0.110529 1.76034 -0.557601 0.00828798
2 4
CurveNode1: 2.51004 -0.625237 -0.554563 3.05133 -0.897748 0.240891 1
CurveNode2: 2.28622 -0.729507 -0.543691 2.66429 -1.5555 -0.125591 0
CurveNode3: 2.0347 -0.758353 -0.503628 2.20628 -1.73976 -0.589666 0
CurveNode4: 1.75172 -0.794524 -0.49784 1.8558 -1.76296 -0.271347 0
3 4
CurveHandle1: -0.940804 0.758812 0.433085 -0.493488 -0.00422765 0.8996
CurveHandle2: -1.42893 0.522248 0.21973 -0.867988 -0.188116 0.644846
CurveHandle3: -0.940246 -0.422196 0.378155 -0.227069 -0.292372 -0.3107
CurveHandle4: -2.83391 -0.847307 0.322513 -2.57244 -1.69312 0.787516
```

Figure 7.9: Example of .STV file. We advise you to use this file format.

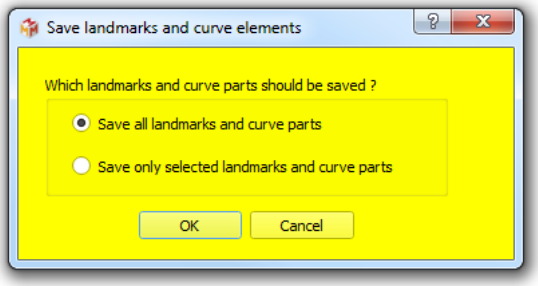


Figure 7.10: Save STV window.

7.4.1 Open MorphoDig Landmark/Curve File (STV)

STV file contain lists of "normal", "target" landmarks (but may also contain lists of "curve node" and "curve handle" landmarks).

7.4.2 Open Landmarks

Landmarks (.VER or .LMK) opened using this option will be put in the “normal" landmark list (light green landmarks)

7.4.3 Open Target Landmarks

Landmarks opened using this option will be put in the “target" landmark list (yellow transparent landmarks)

7.4.4 Save MorphoDig Landmark/Curve File (STV)

You may decide whether you wish to save all (normal, target, curve node, curve handle) landmarks or only selected ones (see Fig. 7.10 p.55).

7.4.5 Save Landmarks

You may decide whether you wish to save only selected “normal" landmarks or all selected and unselected “normal" landmarks (the yellow transparent ones) in .VER or .LMK format (see Fig. 7.11 p.56). . The “Landmarks" chapter (chapter 10 p.89) and the tutorial “working with landmarks" contain further information regarding landmark digitization with MorphoDig.

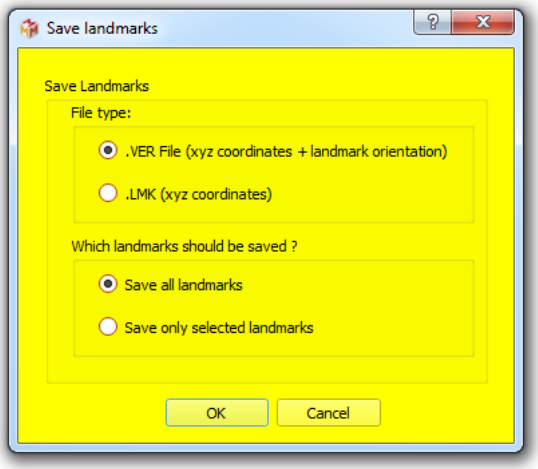


Figure 7.11: Save Landmark Window. Used to save either "Normal", "Target" landmarks (but not both at the same time). This windows is also used to save curve nodes or curve handles in .LMK and .VER format (but also not both at the same time).

7.4.6 Save Target Landmarks

You may decide whether you wish to save only selected “target" landmarks or all selected and unselected “target" landmarks (the yellow transparent ones) in .VER or .LMK format (see Fig. 7.11 p.56). The “Landmarks" chapter (chapter 10 p.89) and the tutorial “working with landmarks" contain further information regarding landmark digitization with MorphoDig

7.5 Curves

3D Curves (series of 3D cubic Bezier curves) are constructed in MorphoDig using 2 series of landmarks : a series of N “curve node" landmarks, and a series of N “curve handle". Curve nodes and handles can be set on surfaces by pressing “L" + left mouse click. In the “curve nodes" landmark mode (button active), pressing “L" + left mouse click results in the creation a “curve node" landmark (a dark green for a curve starting node, and then light red ones). In the “curve handles" landmark mode (button active), pressing “L" + left mouse click will create a “target" landmark (a violet transparent one).

Selected curve nodes/handles can be reordered using the following buttons. Pressing “" will (try to) increase the number of all selected curve nodes/handles, while pressing ““"" will (try to) decrease curve nodes/handles number, respectively.

3D Bezier curves passing through all curve nodes are draw green when no curve node/handle belonging to the curve segment is selected. Curves are drawn red when at least one curve

node/handle involved in the curve segment is selected. Two different cases are considered:

- Case 1: the numbers of “curve node" and “curve handle" landmarks differ. In that case, a curve is a series of lines passing through “curve node" landmarks.
- Case 2: the numbers of “curve node" and “curve handle" landmarks are equal. In that case, a curve is a series of cubic Bezier curves passing through “normal" landmarks. For a given set of 2 “curve node" consecutive landmarks (Ln and Ln+1) and their associated “curve handles" (Hn and Hn+1), a mirror image of Hn+1 relative to Ln+1 (H’n+1) is constructed. The Bezier curve involving Ln, Ln+1, Hn and Hn+1 starts from Ln, going toward Hn, and arrives at Ln+1 coming from the direction of H’n+1.

The explicit form of the curve is :

$$B(t) = (1-t)^3 Ln + 3(1-t)^3 tHn + 3(1-t)t^2 H'n + 1 + t^3 Ln + 1, t \in [0,1] \qquad (7.1)$$

In order to be able to digitize several curves using a given set of curve node and curve handle landmarks, “curve node" landmarks curves can be given 4 flags (see section 10.6 p.92 “Landmarks → selected curve node and handle landmarks" for further details):
Flag “0" : node is within a given curve segment (drawn “red").
Flag “1" : node is the starting point of a curve segment (drawn dark “green").
Flag “2" : node is placed inside the curve, and is a curve “milestone" (drawn blue) .
Flag “3" : node is placed inside the curve, and should be connected to the preceding curve segment starting point. When curve node "N" is flagged that way, curve node "N+1" will be set automatically as a curve segment starting point.
Flag “2" is used to decompose a given curve segment into curve sub-segments. By default, a curve comprises 1 segment.
Flag “3" is used to close a curve (by default, curves are open).
3D curves are loaded and saved into either .STV (more generic) or .CUR (more specific) files. CUR files which contain a series of lines, each line being constructed the following way: name (without space or tab character), curve “node" landmark coordinates, curve “handle" coordinates, flag. In the example shown below, 4 curves are defined :
- an open curve starting from curve node 1 and ending at landmark 7
- a closed curve involving landmarks 8 to 12
- an open curve involving landmarks 13 to 20
- a closed curve involving landmarks 21 to 26

These four curves contain only one sub-segment (no curve milestone was set within those 4 curves). Note that each name does not need to be of the form “CurvePart"+ number. Meanwhile, the name should not hold space or tab characters.

7.5.1 Open Curve (.CUR)

This menu allows the user to load a .CUR file.

```
CurvePart1: 1.30427 -0.948592 -1.93916 1.17336 -1.02898 -2.48304 1
CurvePart2: 1.84539 -2.83943 -2.55712 2.29792 -3.36094 -2.58842 0
CurvePart3: 3.47189 -3.33133 -1.65702 3.77237 -3.15679 -1.16892 0
CurvePart4: 3.26581 -2.10054 -0.418188 2.80311 -1.48064 -0.382995 0
CurvePart5: 2.29388 -1.34609 -1.30233 2.21248 -1.39508 -2.01449 0
CurvePart6: 3.01517 -2.23598 -2.06812 3.4652 -2.4156 -2.03041 0
CurvePart7: 3.76906 -2.2109 -1.4728 3.89444 -2.02446 -1.14752 0
CurvePart8: 3.41419 -1.40082 -1.09418 3.15409 -1.15208 -1.18094 1
CurvePart9: 2.9747 -1.26119 -1.7546 2.9522 -1.43295 -1.99765 0
CurvePart10: 0.918546 0.527479 -0.053347 1.02382 0.817934 -0.170968 0
CurvePart11: 0.866349 1.54776 -0.002301 0.500021 1.84069 -0.04394 0
CurvePart12: -0.697142 1.81948 -0.236996 -1.54527 1.65763 -0.295332 3
CurvePart13: -1.82464 0.592449 -0.117302 -1.83245 0.001632 -0.094651 1
CurvePart14: -1.10252 -0.644453 -0.128808 -0.616091 -0.87967 -0.152128 0
CurvePart15: 0.202059 -0.506594 -0.206101 0.588223 -0.307053 -0.160052 0
CurvePart16: 1.54286 0.057252 0.69468 1.56572 0.270357 1.03118 0
CurvePart17: 1.6504 0.741464 1.4096 1.43427 0.708235 1.79504 0
CurvePart18: 0.557936 -0.100166 2.43925 -0.050625 -0.557494 2.74936 0
CurvePart19: -0.548933 -1.01313 2.26995 -0.861457 -1.22365 2.01447 0
CurvePart20: -1.04871 -1.18664 1.0867 -0.844756 -1.23524 0.722771 0
CurvePart21: -0.348136 -1.3282 0.158734 -0.174153 -1.36476 0.031035 1
CurvePart22: 0.472218 -0.816727 -0.014821 0.678092 -0.657544 0.046628 0
CurvePart23: -0.606305 -1.30496 -1.25154 -0.702298 -1.24359 -1.55922 0
CurvePart24: -1.1233 -0.870954 -1.90743 -1.43065 -0.633659 -1.94748 0
CurvePart25: -1.91427 -0.246833 -1.5827 -2.31769 0.067204 -1.22809 0
CurvePart26: -2.53891 0.288045 0.043388 -2.53802 0.235752 0.611706 3
```

Figure 7.12: Example of .CUR file

7.5.2 Open MorphoDig Landmark/Curve file (.STV)

This menu allows the user to load a STV file, as STF files can contain lists of curve node and curve handle landmarks (see for instance Fig. 7.9 p.54).

7.5.3 Open Curve Node Landmarks

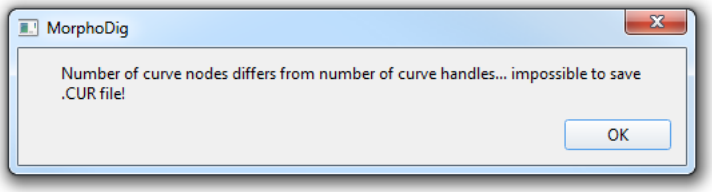
This menu allows the user place the content of a VER or LMK file inside the list of curve node landmarks.

7.5.4 Open Curve Handle Landmarks

This menu allows the user place the content of a VER or LMK file inside the list of curve handle landmarks.

7.5.5 Save .CUR file

This menu allows the user to save current landmarks and curve handles as a .CUR file. This action is only allowed if the number of “curve node" landmarks and “curve handle" landmarks is the same. If not, the following message appears:



If you are in the process of digitizing curves (and do not have achieved to digitize the same number of “normal" and “target" landmarks and wish to save the current state of your work, you may decide to save curve node and target landmarks within a STV file instead (see next section).

7.5.6 Save MorphoDig Landmark/Curve File (STV)

You may decide whether you wish to save all curve node and curve handle landmarks or only selected ones (see Fig. 7.10 p.55). Using this option will also save "normal" and "target" landmark lists.

7.5.7 Save Curve Node Landmarks

Curve node landmarks can be saved inside a LMK or VER file using this option. You may decide whether you wish to save only selected “curve node" landmarks or all selected and unselected “curve node" landmarks in .VER or .LMK format (see Fig. 7.11 p.56).

7.5.8 Save Curve Handle Landmarks

Curve handle landmarks can be saved inside a LMK or VER file using this option. You may decide whether you wish to save only selected “curve handle" landmarks or all selected and unselected “curve handle" landmarks in .VER or .LMK format (see Fig. 7.11 p.56).

7.5.9 Export Curves as Landmark file

Curves can be transformed in a series of equidistant landmarks using this option. The curve decimation window appears. Each curve **segment** is saved as a number of equidistant landmarks.
Be aware that there are two ways to define curve **segments** in MorphoDig :
- by setting curve node "starting points" , in order to create **independent** curve segments (see section 10.6.2 p.92, and a practical example Fig. 10.7 p. 94)
- by setting curve node "milestones", in order to create **contiguous** curve segments (see section 10.6.4 p.96, and a practical example Fig. 10.9 p. 95).
You may also decide to save normal and target landmarks information inside the VER or LMK output. This is useful when you need to use both type 1 landmarks (normal and target landmarks) and semi-landmards (3D curves) in Geometric Morphometric analyses.
In the present example, each curve/ curve segment is saved as 20 equidistant landmarks.

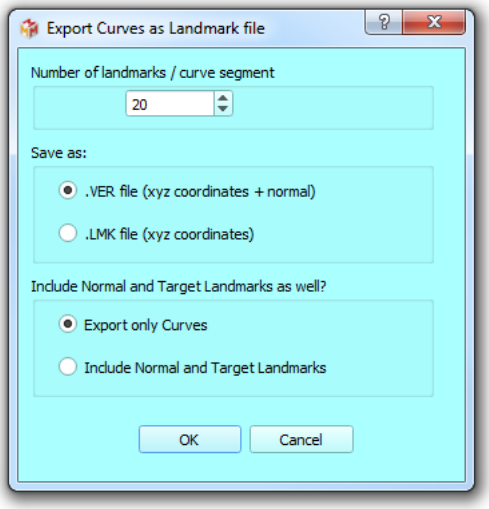


Figure 7.13: Export curves as landmark file window

7.5.10 Save curve infos (length per curve segment)

Each curve **segment** length can be saved as a .txt file using this option. As stated in the preceding section, we want to insist on the fact that there are two ways to define curve **segments** in MorphoDig :
- by setting curve node "starting points" , in order to create **independent** curve segments (see section 10.6.2 p.92, and a practical example Fig. 10.7 p. 94)
- by setting curve node "milestones", in order to create **contiguous** curve segments (see section 10.6.4 p.96, and a practical example Fig. 10.9 p. 95).

The “Landmarks" chapter (chapter 10 p.89) and the tutorial “Working with curves" contain further important information regarding curve digitization with MorphoDig.

```
Curve_segment_1_length(mm):10.7182
Curve_segment_2_length(mm):12.5473
Curve_segment_3_length(mm):10.4387
Curve_segment_4_length(mm):9.65491
```

Figure 7.14: Example of curve info file

```
Premaxilla
2.993259 -0.213141 -0.175908 3.349062 -0.902149 0.455496 1.000000 1.000000 1.000000 0.000000
Vomer
2.804241 0.314822 -0.089269 3.659665 0.784189 -0.308230 1.000000 1.000000 0.666667 0.000000
Dentary
2.361118 0.250747 -0.402550 2.448524 0.362454 -1.392440 1.000000 0.000000 0.666667 0.000000
Dermopalatine
2.242324 0.572392 0.056026 2.197675 1.003808 0.957073 1.000000 0.666667 0.000000 1.000000
Maxilla
1.673053 -0.704019 -0.137551 1.431000 -0.920334 0.808292 1.000000 1.000000 0.482353 1.000000
Ectopterygoid
1.512049 -0.437096 -0.115018 1.215033 -0.451175 0.839751 1.000000 0.666667 0.000000 1.000000
Frontal
0.665411 0.543348 0.739945 0.875703 0.834776 1.673137 1.000000 1.000000 0.000000 0.000000
Parasphenoid
-0.509335 -0.104936 -0.140395 -0.591932 -0.001493 -1.131595 1.000000 1.000000 1.000000 0.000000
Otolith
-1.380154 0.592150 0.273919 -1.045224 0.682863 1.211785 1.000000 1.000000 0.666667 0.494118
Supracleithrum
-2.469155 -0.298203 0.924624 -2.593424 0.688945 1.025103 1.000000 0.000000 0.666667 1.000000
Cleithrum
-2.814436 0.567808 -0.124991 -3.479043 1.298076 -0.283128 1.000000 0.000000 0.666667 1.000000
```

Figure 7.15: Example of .FLG file

7.6 Flags

Regarding flags, as stated earlier, one series of "flag landmarks" can be digitized in MorphoDig (button "  " should be pressed). To edit flag label, length and color, select one flag landmark, click on "  " in order to open the "Edit first selected landmark" window (Fig. 6.14 p.36). Pressing ok will update the label, the color and the length associated to the selected flag, which in turn will be unselected.
Flags are saved using the .FLG file format, which consists of n pairs of lines constructed the following way (Fig. 7.15 p.61):
line 2*n: Flag name
line 2*n+1: Flag coordinates, flag orientation, flag length and color.

7.6.1 Load flags

Select a .FLG file using this menu

7.6.2 Save flags

This option saves the current flag landmarks into a .FLG file, regardless their selection status.

7.7 Color maps

- Color maps are used to visualize scalar arrays (=arrays of numbers associated to each vertex) and can be edited interactively by clicking on  , which opens the "Scalar rendering options"

A

```
nr: 1
name: Rainbow
discretize: 0
discretizenr: 256
enableopacity: 1
nc: 6
no: 4
c0: 0 1 0 1
c1: 0.2 0 0 1
c2: 0.4 0 1 1
c3: 0.6 0 1 0
c4: 0.8 1 1 0
c5: 1 1 0 0
op0: 0 0.3
op1: 0.2 0.6
op2: 0.8 0.8
op3: 1 1
```

B

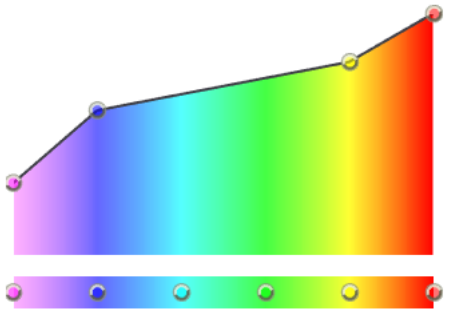
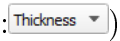


Figure 7.16: A. Example of .MAP file containing one single colormap named "Rainbow", and having 5 color and 4 opacity control points. B. Translation of this colormap inside MorphoDig.

window (see chapter “Scalars" (chapter 11 p.99) and the tutorial “Working with scalars" for further information).
- Scalars become visible when the array display mode button is pressed (), and a scalar array is selected (ex: 

).
Two default color maps exist in MorphoDig, which can be edited, but not deleted : the "Rainbow" the "Black-Red-White" color maps. Any number of additional color maps can be added to the list of existing color maps, and are referred to as "custom color maps".

An example of color map (.MAP) file and how it translates into an actual color/opacity map is shown Fig. 7.16 p.62):

7.7.1 Import color maps (.MAP)

Imports one or several color maps included inside a .MAP file.

7.7.2 Export color maps (.MAP)

This option opens the Export color maps window, in which the active color map, all color maps (the 2 default color maps + all custom color maps), or all custom color maps can be saved inside a .MAP file.

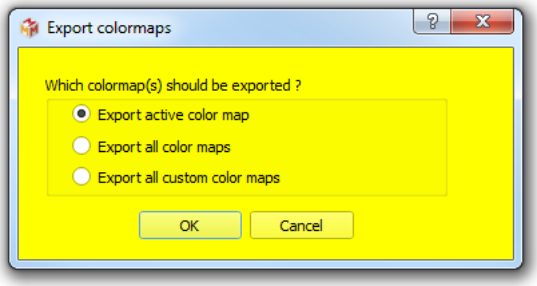


Figure 7.17: Export color maps window.

7.8 Tag maps

- Tag colors and names (=tag maps) can be edited interactively by clicking on , which opens the Tags window (see the chapter “Tags" (chapter 12 p.123) and the tutorial “Working with tags" for further information).
- Tags become visible when the array display mode button is pressed (), and a tag array is selected (ex: ConnectivityTags).

Tag maps consist mostly of a series of combination of a tag name and an associated color. Any number of tag name + color can be defined (25 different names+colors by default, but this number can be interactively increased/decreased).
By default, MorphoDig contains one Tag maps named "TagMap", which can not be deleted, but can be edited as needed. Other tag maps can be added manually to tag map list, and are referred to as "custom tag maps".

Tag map files come in two formats : .TAG and .TGP.

TAG format consists of N pairs of lines, can store one single tag map, and is constructed the following way: line 2*n: Tag name
line 2*n+1: Tag color and transparency

```
exterior
0.658824 0.560784 0.372549  1.000000
cochlea
0.000000 0.552941 0.686275  1.000000
lateral canal
0.000000 0.000000 1.000000  1.000000
anterior canal
0.556863 0.031373 0.474510  1.000000
posterior canal
0.827451 0.713726 0.117647  1.000000
vestibule
0.388235 0.125490 0.800000  1.000000
ovale window
0.000000 0.521569 0.145098  1.000000
round window
1.000000 0.474510 0.000000  1.000000
common crus
0.290196 0.521569 0.000000  1.000000

(etc...)
```

Figure 7.18: Example of .TAG file

TGP format differs from TAG in that it can store several tag maps and gives a name to each tag map. An example of TGP file containing 2 tag maps is shown aside.
line 2*n: Tag name
line 2*n+1: Tag color and transparency

```
nr: 2
name: TagMap
nt: 3
t0: Exterior
c0: 0.701961 0.8 0.8 0.8
t1: Tag1
c1: 0.639216 0 0.8 0.8
t2: Tag2
c2: 1 0 0 0.8
name: Custom_TagMap
nt: 4
t0: Exterior
c0: 0.701961 0.8 0.8 0.8
t1: Mx
c1: 0.666667 0.666667 0 1
t2: Md
c2: 1 0 0 0.8
t3: Teeth
c3: 0 1 0 0.8
```

Figure 7.19: Example of .TGP file containing two tag maps. The first contains 3 entries, the second 4 entries.

7.8.1 Import tag maps (.TGP or .TAG)

Imports one tag map (.TAG or .TGP file) or several tag maps (.TGP file). Then open the tag window (click on ) : Tag labels, colors and transparencies should have been updated. In the "Tags window", you can switch between different tag maps with the combo box "Choose tag map".

7.8.2 Export tag maps (.TGP or .TAG)

This option opens the Export tag maps window, in which the active tag map, all tag maps (= default tag map + all custom tag maps), or all custom tag maps can be saved inside either a .TAG file (if only one tag map should be saved) or inside a .TGP file (one or several tag maps).

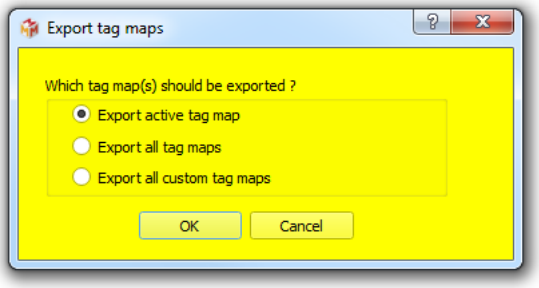


Figure 7.20: Export tag maps window.

7.9 Orientation helper labels

The coordinate system orientation helper labels can be saved into ".ORI" files, which are .txt files containing 6 lines, 1 for each axis.

7.9.1 Open Orientation labels

Select a .ORI file using this menu. Then open the orientation labels window window (Edit→Edit Orientation labels) : the 6 orientation labels should have been updated.

7.9.2 Save Orientation labels

This option saves the current state of orientation labels in a .ORI file.

7.10 Measurements

7.10.1 Save area, volume, triangle number and vertex number of selected surfaces

Surface area, volume, triangle number and vertex number of all selected surface objects can be saved in a .txt file using this option.
Note: surface objects should be closed in order to provide a correct estimation of object volume.

7.10.2 Complexity: save normalized shape index, area and volume of selected surfaces

Surface normalized shape index (NSI) can be viewed as a measurement of surface "complexity" (= how much area can be enclosed within a given volume). Surface normalized shape index (NSI), surface area and surface volume of all selected surface objects can be saved in a .txt file using this option.
For a given surface of area "SA" and of volume "V", "NSI" is a measurement of how much the ratio between surface area and surface volume differs from that of a sphere. It is defined as:

$$NSI = F\frac{\sqrt[2]{(SA)}}{\sqrt[3]{(V)}} \tag{7.2}$$

where F is a constant defined so that perfectly spherical 3D surfaces express a NSI index equal to 1, and higher values for non spherical shapes:

$$F = \frac{\sqrt[3]{(\frac{4}{3}\pi)}}{2\sqrt[2]{(\pi)}} \tag{7.3}$$

7.10.3 Complexity: save convex hull area ratio and convex hull normalized shape index of selected surfaces (warning: slow).

Surface normalized shape index (NSI, see preceding section) may not offer a good estimate of "global shape complexity" of surfaces, especially when surfaces' shape differ very much from that of a sphere (very flat objects). Other metrics, such as Convex hull area ratio (ChAR), or convex hull normalized shape index (ChNSI), may be better proxies for a global measurements of shape complexity for whole surfaces (= how much area can be enclosed within a given volume). ChAR, ChNSI and surface area, surface volume, convex hull, convex hull volume and surface normalized shape index (NSI: defined in the preceding section) of all selected surface objects can be saved in a .txt file using this option.

ChAR

For a given surface of area "SA", ChAR is the ratio between "SA" and the surface area of the convex hull of that same surface (ChSA). Its aim is to compare the surface area of a surface with that of that of a "wrapping bag" that would enclose that surface. It is defined as:

$$ChAR = \frac{SA}{ChSA} \tag{7.4}$$

Perfectly spherical 3D surfaces express a ChSA index equal to 1 while, while hihly folded structures (such as turbinates) usually express values higher than 1. Shapes spread in space and containing a lot of holes may express ChSA values smaller than 1.

ChNSI

For a given surface of area "SA" and of volume "V", "ChNSI" measures how much surface area "SA" can be enclosed within the volume of the convex hull ("ChV") computed for that same surface It is defined as:

$$ChNSI = F\frac{\sqrt[2]{(SA)}}{\sqrt[3]{(ChV)}} \tag{7.5}$$

where F is a constant defined so that perfectly spherical 3D surfaces express a ChNSI index equal to 1 :

$$F = \frac{\sqrt[3]{(\frac{4}{3}\pi)}}{2\sqrt[2]{(\pi)}} \tag{7.6}$$

Perfectly spherical 3D surfaces express a ChNSI index equal to 1. Hihly folded structures (such as turbinates) usually express ChNSI values higher than 1. Shapes spread in space and containing a lot of holes may express ChSA values smaller than 1.

7.10.4 Save size measurements (max length in xyz direction etc.) of all selected surfaces.

Depending on your biological scientific question, you may need different measurements of the "size" of your structure of interes. This menu provides different "size metrics" for 3D surfaces which can be relevant.

Bounding box length

Bounding box length gives an idea of the maximal length of a 3D surface. However, this metrics depends on the orientation of the surface, so be careful when using it.

Mean and maximal distances to centroid

For a given surface containing N vertices, the centroid is defined as the average x,y,z coordinates of all N vertices. The mean distance of all N vertices to the centroid, and the maximal distance for all N vertices to the centroid give information regarding the expansion in 3D space of your structure.

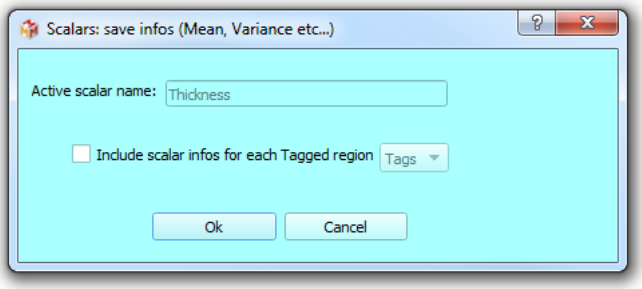


Figure 7.21: "Scalars: save infos" dialog.

PC1, PC2, PC3 length and average of PC1, PC2, PC3

A better alternative to the bounding box length metrics is as follows: for a given surface, MorphoDig computes a Principal Component Analysis (PCA) of all 3D vertex coordinates. Then all vertices are projected on Principal component 1 (PC1), on PC2 and on PC3. The difference between the maximal projection scores and minimal projection scores on a given PC give the length of your structure of interest along that axis. PC1 length is a measurement of the maximal length of your structure. PC2 and PC3 lengths are measurements of the maximal width and height of your structure. PC1, PC2 and PC3 length do not depend on 3D surface orientation and are in that sense much reliable metrics than bounding box length.

7.10.5 Save active scalar infos (mean, median, variance ...) of all selected surfaces.

In order to use this option, the array display mode button must be pressed ( ), and a scalar array must also be selected selected (ex: Thickness).
This option opens the "Scalars: save infos" dialog (see Fig. 7.21 68), and saves within a .txt file, and for all selected surfaces, the mean, the median, and the variance of the selected scalar. If the "Include scalar infos for each Tagged region" option is checked and if a the selected tag array exists for the involved surfaces, then for each tagged region, the mean, the median and the variance of the currently active scalar is also computed.

7.10.6 Save scalar values of first selected surface.

This option saves within a .txt file, and ONLY for the first selected surface containing N vertices, all scalar values for all vertices. Ex: if a given surface contains 3 scalars (ex: thickness, curvature, complexity), the .txt file will contain the values of the 3 scalar arrays for all N vertices.

Chapter 8

Menu Edit

Contents

8.1 Edit color options

The color options windows (Fig. 8.1 p.70) makes it possible to edit the default solid color of newly opened surfaces, the background color of the upper part of the 3D window, and the background color of the lower part of the 3D window. Default color options can be reinitialized by pressing on the "Reinitialize" button.

8.2 Edit size unit, grid spacing and scale

The "Size unit, grid spacing and scale" window (Fig. 8.2 p.70) contains the following sections:

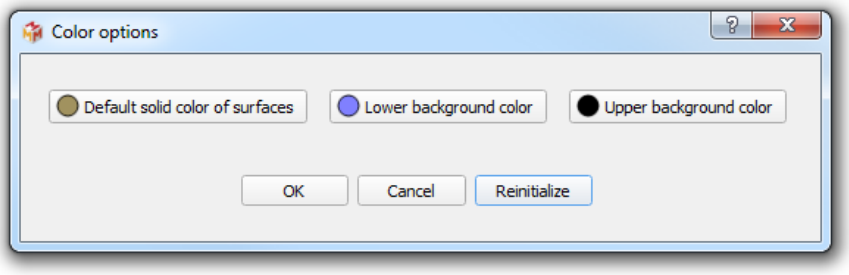


Figure 8.1: Color options window.

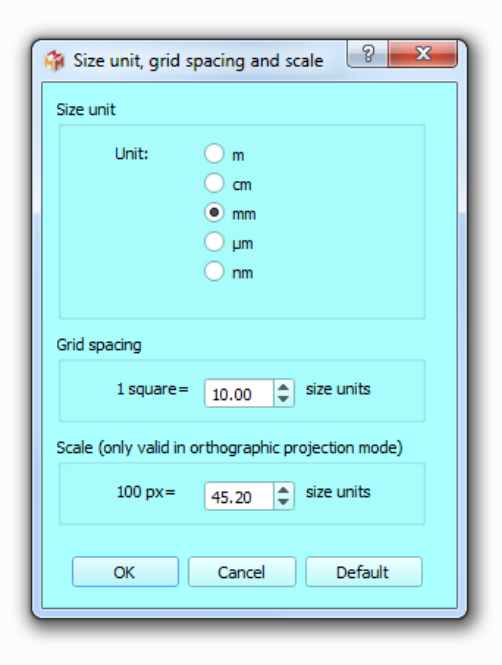


Figure 8.2: Size unit, grid spacing and scale window.

8.2.1 Size unit

MorphoDig's default size unit is "mm". However you can change the currently used size unit of your 3D surfaces. Indeed, different 3D surfaces may contain 3D x,y,z coordinates coded in different size units. Regarding 3D surfaces of biological models, one commonly used size-unit is "mm". But 3D surfaces containing 3D x,y,z coordinates expressed in "µm" are also quite common. Be careful: MorphoDig will not be able to render at the correct scale two 3D surfaces using different 3D coordinates size units: if you have a first 3D surfaces expressed in "mm", and a second 3D surface expressed in "µm", the second surface will be rendered around 1000 times larger than the first one, and there is currently no straightforward way to detect which size unit is used for a given 3D surface.

8.2.2 Grid spacing

You can change the rendering size of the grid. By default 1 grid is 10 mm. But depending on the size of your biological structure of interest, you may find it convenient to adjust grid display size to that of your 3D objects.

8.2.3 Scale (only valid in orthographic projection mode)

As stated earlier, you can switch between orthographic and perspective projection mode by pressing the "  " or "  " toggle button. In this section, you can adjust the display scale (=zoom) in order that 100 pixels on the screen translate into a desired number of size units in real world. This option is extremely useful to construct scale bars on scientific illustrations using software such as Gimp of Photoshop. For instance, if you set the display scale in order to have 100 pixels = 1 mm and then make a screenshot of your 3D objects in MorphoDig, if you want to place a scale bar of 5mm, you will simply have to create and draw a 500 pixel-wide rectangle over it.

8.3 Landmark and flag rendering options

The landmark and flag options window (see Fig. 8.3 p.72) contains the following sections:

8.3.1 Landmarks rendering

"Normal", "Target", "Curve node" and "Curve handle" landmarks can be drawn as spheres or as arrows (composed of a stick and a sphere). Landmark display size can be chosen to be adjusted automatically (default behaviour) and furthermore adjusted using an adjustment scale factor. Fig. 8.4 p.73 shows the effect of the modification of this adjustment scale factor. Alternatively, you may decide to set manually landmark to a fixed rendering size.

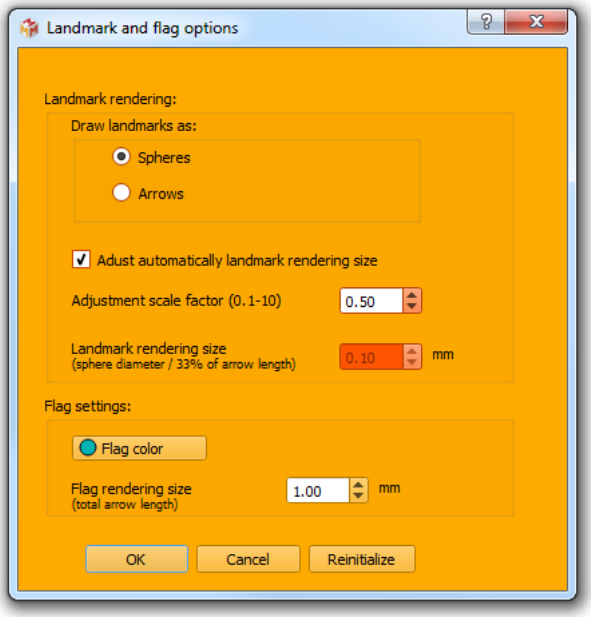


Figure 8.3: Landmark and flag options window.

8.3.2 Flags settings

Flag rendering size and flag color settings can be defined in this subsection. Once placed on a surface and selected, the color, the length and the label of the flag can be changed by pressing "flag edit". This opens the "Edit first selected flag" window (Fig. 6.14 p.36).

8.4 Edit orientation labels

The orientation labels of the orientation helper can be modified, or reinitialized to their default values via this menu (Fig. 8.5 p.74).

As stated earlier, you can press  to show/hide the coordinate system orientation helper lying on the bottom left corner of the main 3D window.

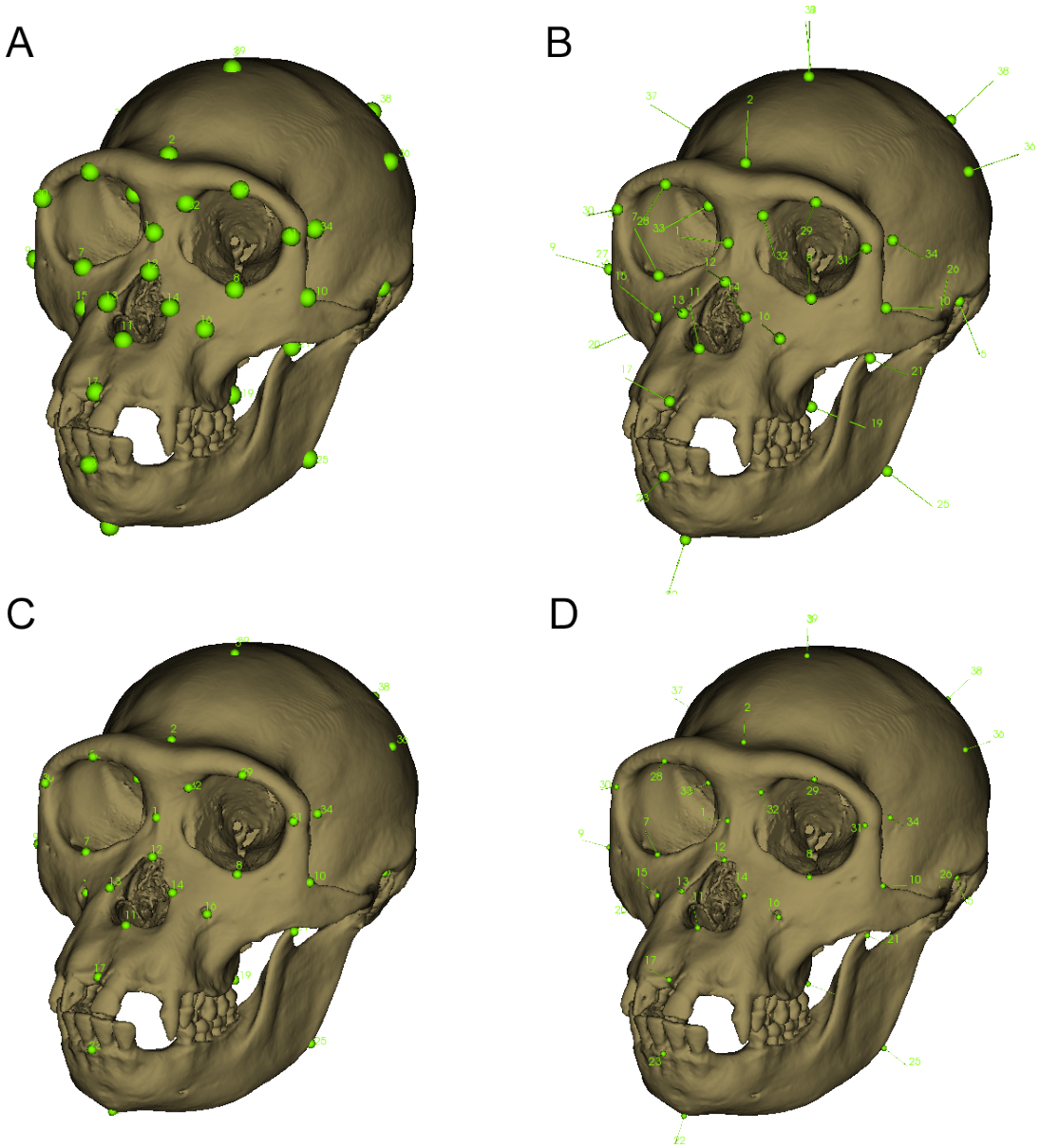


Figure 8.4: Example of the effect of different rendering options for the same set of landmarks (specimen: cranium and mandible of the type specimen of *Pan paniscus*). A: Landmarks are drawn as spheres, rendering size is adjusted automatically, adjustment factor = 1. B: Landmarks are drawn as arrows, rendering size is adjusted automatically, adjustment factor = 1. C: Landmarks are drawn as spheres, rendering size is adjusted automatically, adjustment factor=0.5. D: Landmarks are drawn as arrows, rendering size is adjusted automatically, adjustment factor=0.5.

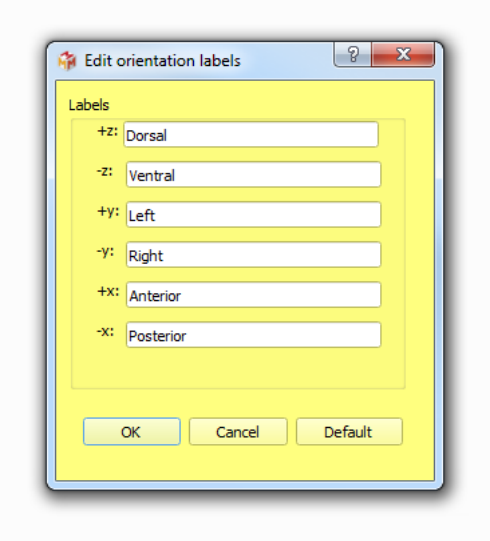


Figure 8.5: Edit orientation labels window.

Chapter 9

Menu Surfaces

Contents

In most cases, these function only apply on selected surfaces, so, as a prerequisite, you almost always need to select surfaces in order to use what is described below.

Figure 9.2: Smoothing surfaces. Left: example of original input surface (left inner ear of *Galago moholi*). Right: resulting output surface after 100 iterations using a relaxation factor of 0.1.

9.1 Structure modification

9.1.1 Smooth each selected surface

This option uses vtkSmoothPolyDataFilter.You may smooth an input selected surface using this option (see Fig. 9.2). A number of iteration and a relaxation factor are required.
See vtkSmoothPolyDataFilter documentation for further information regarding this option.

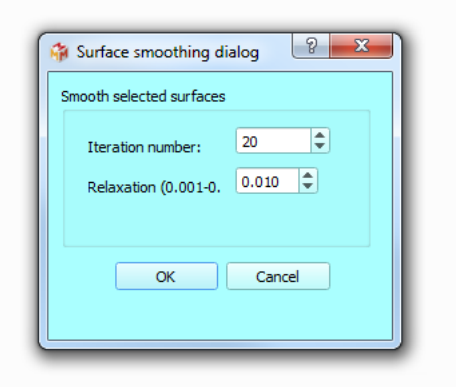


Figure 9.1: Surface smoothing dialog

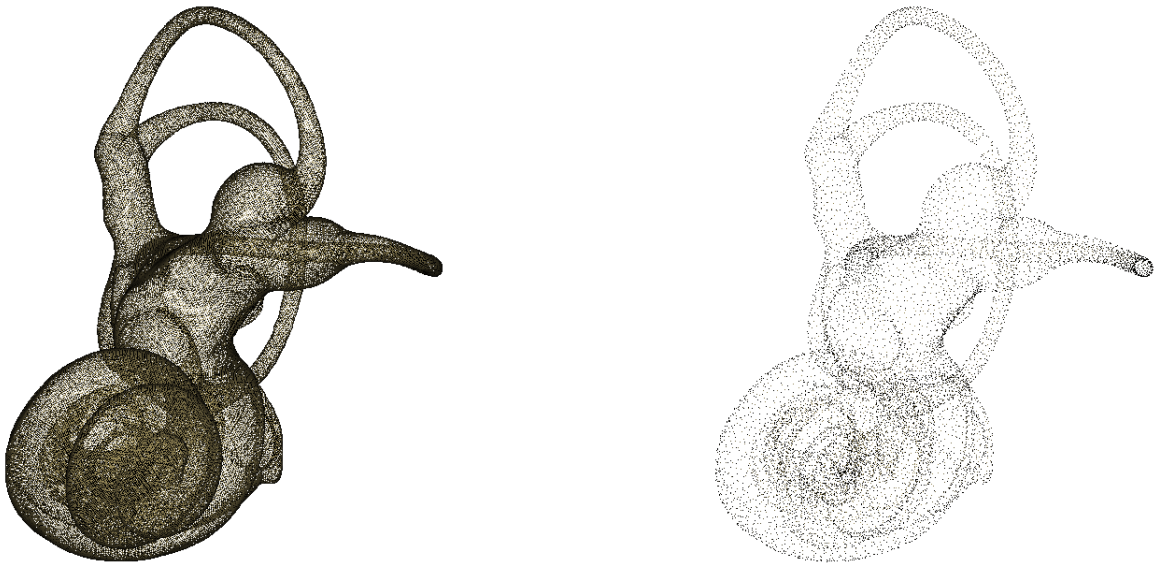


Figure 9.4: Surface decimation. Left: point cloud rendering of input surface (left inner ear of *Galago moholi*). Right: point cloud rendering of resulting output (vtkQuadricDecimation filter, decimation factor: 95%).

9.1.2 Decimate each selected surface

This option uses vtkDecimatePro and vtkQuadricDecimation filters. Requirements : to use mesh decimation, a selected surface is required (see for instance Fig. 9.4). See vtkDecimatePro and vtkQuadricDecimation documentations for further information regarding mesh decimation.

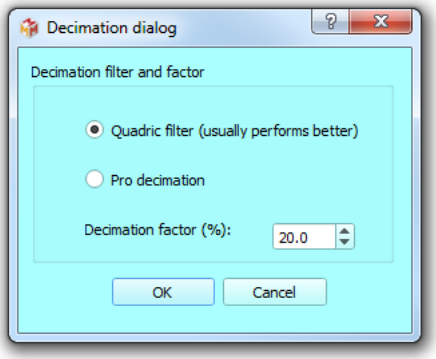


Figure 9.3: Decimate window

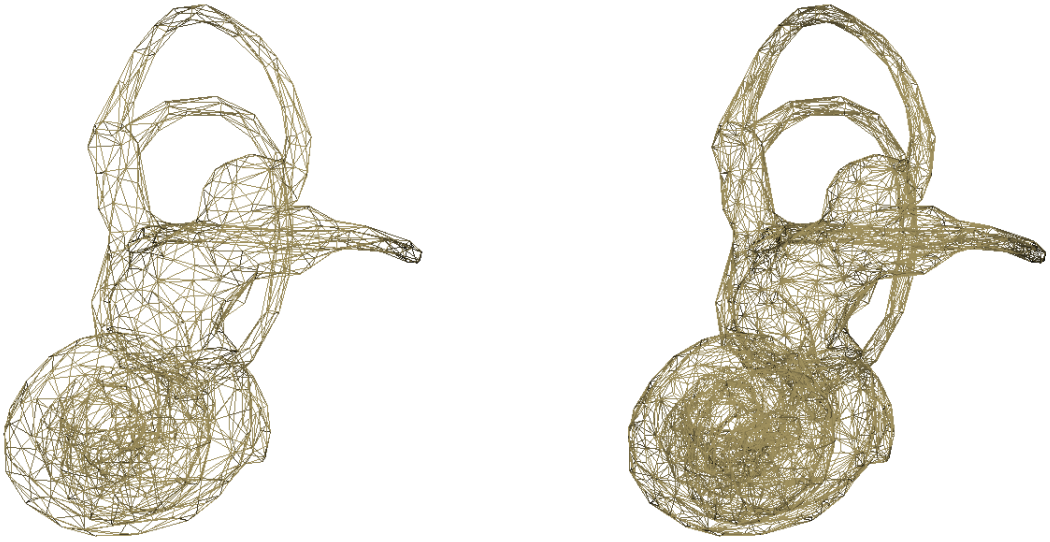


Figure 9.6: Surface densification. Left: wireframe rendering of input surface (left inner ear of *Galago moholi*). Right: wireframe rendering of resulting output (number of subdivisions: 1).

9.1.3 Densify each selected surface

This option uses vtkDensifyPolyData filter. Requirements : to use mesh densification, a selected surface is required (see for instance Fig. 9.6). Note that mesh decimation can become extremely slow when using number of subdivisions larger than 1. See vtkDensifyPolyData documentation for further information regarding mesh densification.

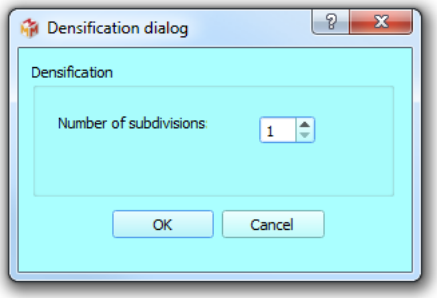


Figure 9.5: Densify dialog

Figure 9.8: Filling holes. Left: input surface: left inner ear of *Galago moholi* having a hole in the round window. Right: resulting surface output.

9.1.4 Fill holes of each selected surface

This option uses vtkFillHolesFilter. Requirements : to use mesh hole filling, a selected surface is required (see for instance Fig. 9.8). See vtkFillHolesFilter documentation for further information regarding hole filling.

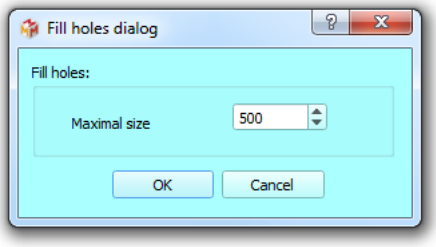


Figure 9.7: Fill holes window

9.1.5 TPS deformation of each selected surface

This option uses vtkThinPlateSplineTransform filter. Requirements : to use TPS deformation, a selected surface, a series of “n" normal landmarks and a series of “n" target landmarks (n>3) are needed. “Normal" landmarks are usually placed on the original selected input surface, whereas “target" landmarks are placed at a location in 3D space which will drive the TPS deformation (see Fig. 9.10). See vtkThinPlateSplineTransform documentation for further information regarding TPS deformation.

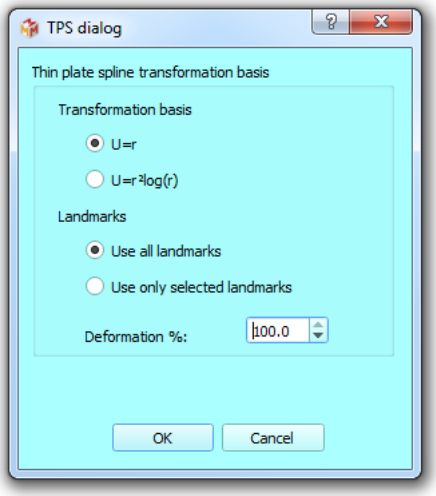


Figure 9.9: TPS dialog

9.1.6 Connectivity: separate each selected surface into independent regions

This option uses vtkPolyDataConnectivityFilter. This filter produces a new surface for each non-connected region of the selected input surface. 3D meshes of biological objects sometimes contain a multitude of small and biologically irrelevant independent regions. This “noise" may have multiple origins: low quality of original 3D data, state of preservation of the specimen, threshold used to be able to visualize all relevant structures, etc... In order to extract relevant independent regions, only regions reaching a minimal size (minimal number of triangles) are transformed into new surfaces (see Fig. 9.12). This process may take some time to be completed. All produced surfaces corresponding to independent regions can be manipulated independently.

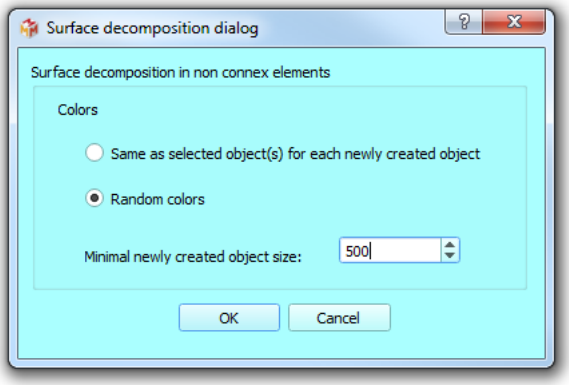


Figure 9.11: Connectivity decomposition dialog

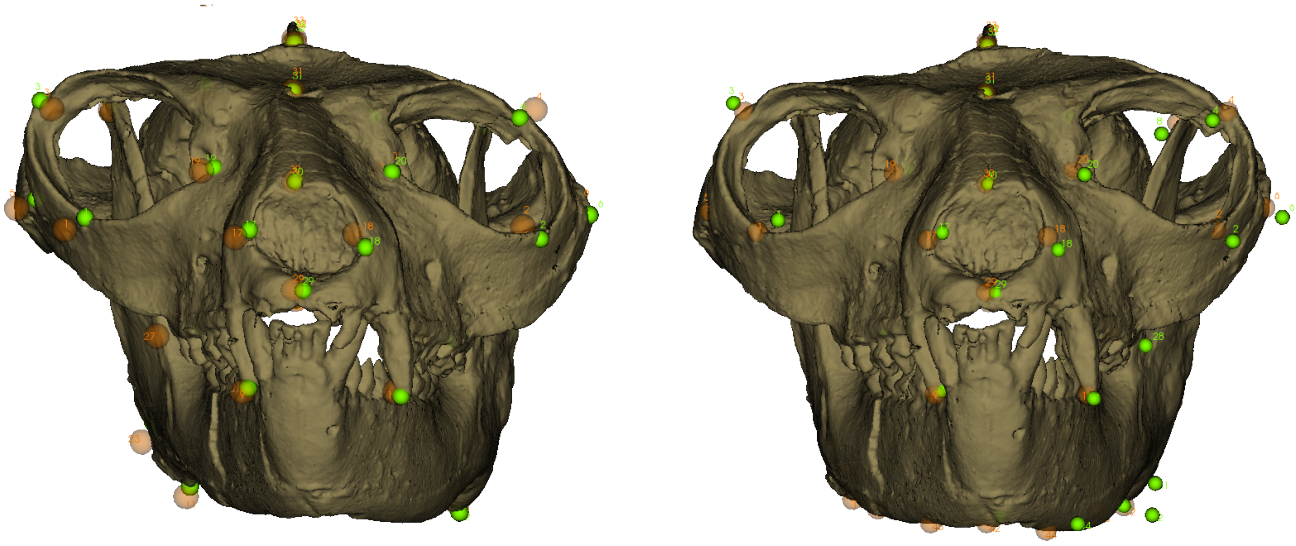


Figure 9.10: Thin plate spline (TPS) transformation. Left: original distorted input surface of the cranium and mandible of *Notharctus tenebrosus* (3D surface obtained from a cast of specimen AMNH 127167 from the American Museum of Natural History, New York City, New York, USA). 46 “normal" landmarks were placed on the surface and 46 “target" landmarks were placed in order to restore bilateral symmetry. Right: resulting output (deformation : 100%). Note that the 46 “target" landmarks are located on the output surface.

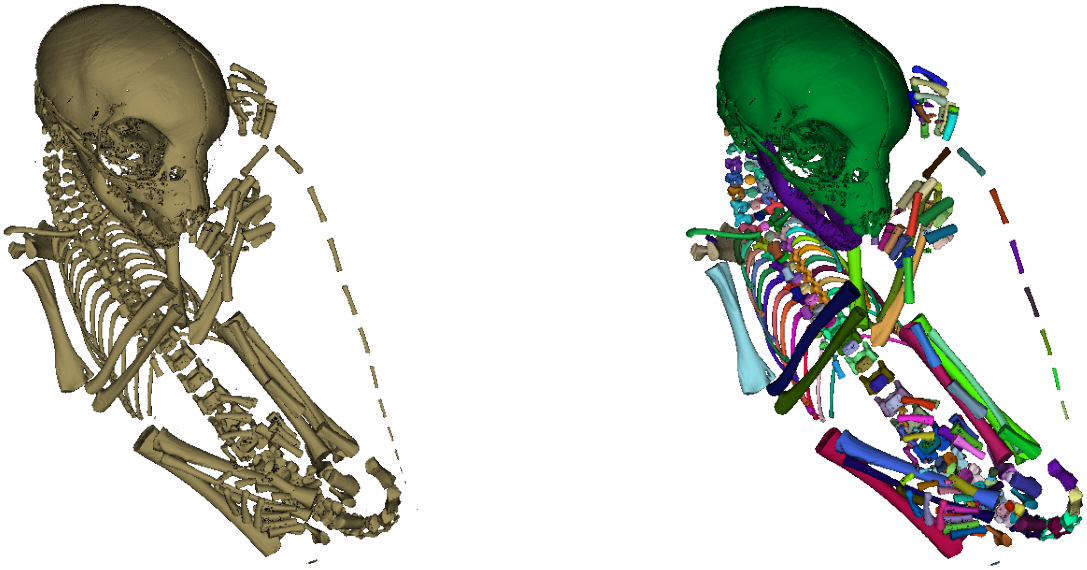


Figure 9.12: Example of surface decomposition into non-connected surfaces. Left: original surface of the skeleton of a newborn *Lemur catta* containing a large number of independent regions of size greater than 500 triangles. Right : Output result (independent surface objects) Filtered surfaces. All surfaces produced using this filter have more than 500 triangles and were given a random color. Note that several bones of the most distal part of the tail are absent, because they contain less than 500 triangles.

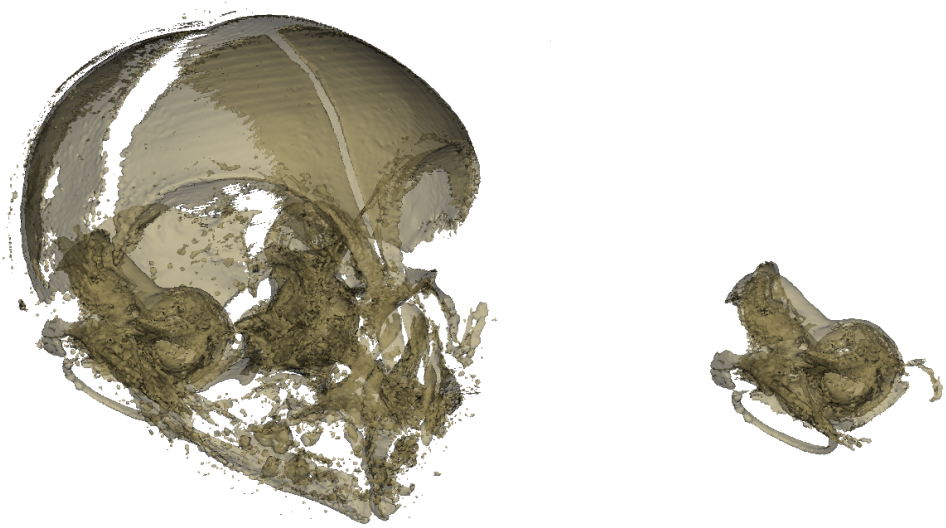


Figure 9.13: Example of surface decomposition in order to keep largest region. Left: original 3D surface representing the skull of a newborn *Tarsius bancanus*. Right: the resulting largest region in terms of triangle number.

9.1.7 Connectivity: keep largest region for each selected surface

This option uses vtkPolyDataConnectivityFilter.This filter produces a new surface for the largest independent region of the selected input surface (see Fig. 9.13).

9.1.8 Invert each selected surface

A given surface can be inverted in order to show inner structures (see Fig. 9.14).

9.1.9 Mirror each selected surface along Y plane

This option uses vtkReflectionFilter, which produces a mirror image of the original selected input mesh (see Fig. 9.15).

9.2 Convex hulls

9.2.1 Create a convex hull for each selected surface

This option uses vtkDelaunay3D.This filter produces convex hull surfaces for each selected surface. Convex hulls can be useful for instance to compute estimations of the total "volume"

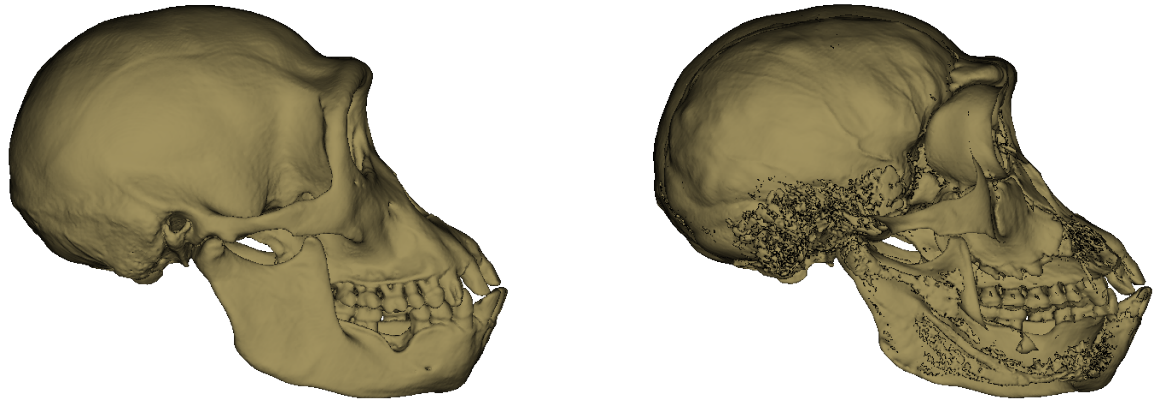


Figure 9.14: Example of surface inversion. Left: original surface of the skull of the type specimen of *Pan paniscus*. Right: backface culling rendering of the same surface after being inverted, revealing inner structures such as the endocranial cavity.

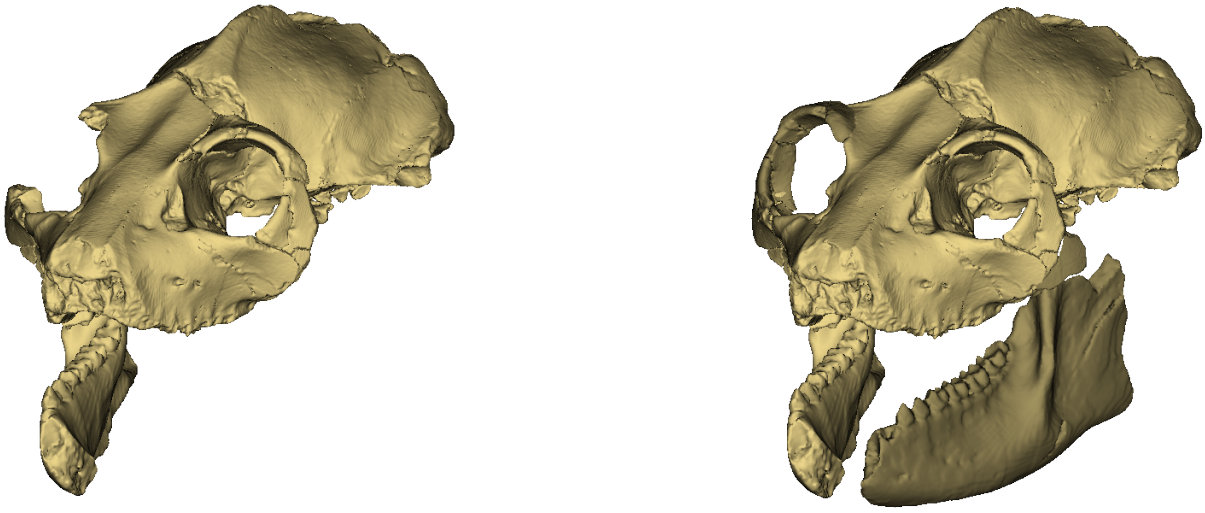


Figure 9.15: Example of fossil restoration (*Palaeolemur betillei* BOR613 specimen from the Musée d'Histoire Naturelle de Bordeaux, France) implicating the production of mirror images of missing parts.

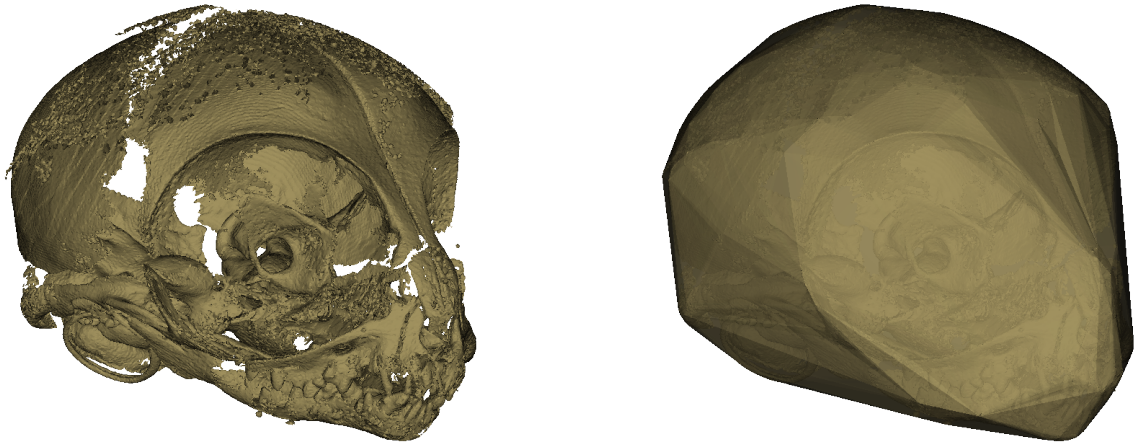


Figure 9.16: Convex hull computation. **Left:** original 3D surface representing the skull of a newborn *Tarsius bancanus*. Bone volume: ~100 mm3. **Right:** the resulting convex hull. Volume of the convex hull of the skull (=proxy for the volume of the "head"): ~3653 mm3.

occupied in space by complex objects containing lots of holes (see Fig. 9.16).

9.2.2 Merge selected surfaces into one single surface

This options makes it possible to merge all selected surfaces into one single surface object.

Figure 9.18: Example of alignment of two mandibles of *Tarsius bancanus*. A: the two surfaces in their initial orientation. B: the two mandibles were roughly manually oriented in the same direction. C: result of the ICP algorithm. The impacted surface (the one which was actually reoriented) is the blue one.

9.3 Surface alignment

This option uses vtkIterativeClosestPoint-Transform filter. This filter makes it possible to align two surfaces using the Iterative Closest Point (ICP) algorithm. In order to get good results, it is strongly advised to orient manually roughly the two surfaces in the same direction before running this algorithm (see for instance Fig. 9.18). See vtkIterativeClosestPointTransform documentation for further information.

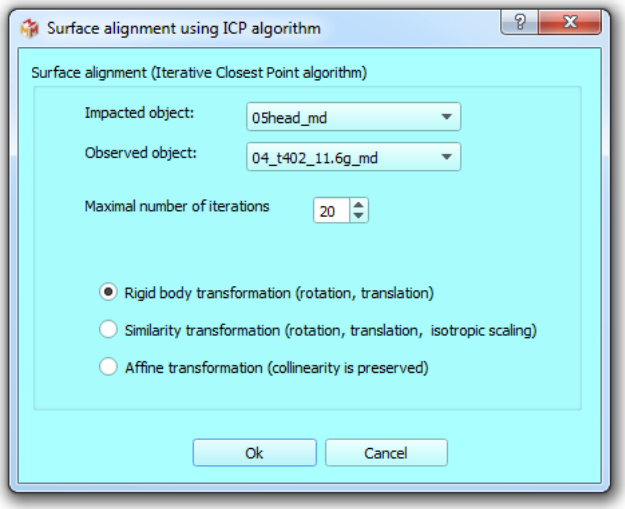


Figure 9.17: Surface alignment dialog

9.4 Rendering modification

9.4.1 Change transparency

All selected actors can be given the same transparency using this option (at least one selected surface is needed).
Please choose a value between 0 and 100. 100 stands for “opaque rendering". 0 stands for “invisible surface".

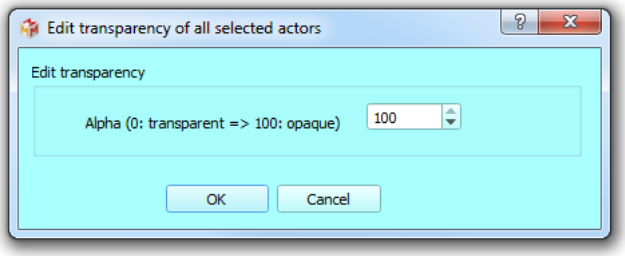


Figure 9.19: Edit transparency dialog

9.4.2 Change object solid color

Selected objects’ solid color can be changed using this option. A set of 13 predefined colors is available via this menu. Alternatively, you can edit object color manually using the “Default solid color of surfaces" control of the “Edit color options" window (menu “Edit →Edit color options").

Chapter 10

Menu Landmarks

Contents

As stated above, landmarks can be set on surfaces by pressing "L" + left mouse click. Several actions can be performed on landmarks.

10.1 Select landmark range

By opening the “select landmark range" window, you can select a given range of landmarks. This option may be useful when you need to save only a specific sub-range of all digitized landmarks.

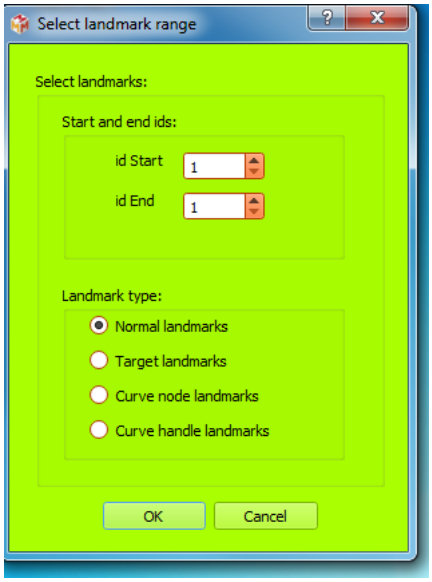


Figure 10.1: Select landmark range window

10.2 Selected landmarks: decrease landmark number (move up in list)

This option will increase (=move up in list, if possible) the landmark number of all selected landmarks. This option can be also activated by clicking on in the "object controls" section of the main window.

10.3 Selected landmarks: increase landmark number (move down in list)

This option will decrease (=move down in list, if possible) the landmark number of all selected landmarks. This option can be also activated by clicking on in the "object controls" section of the main window.

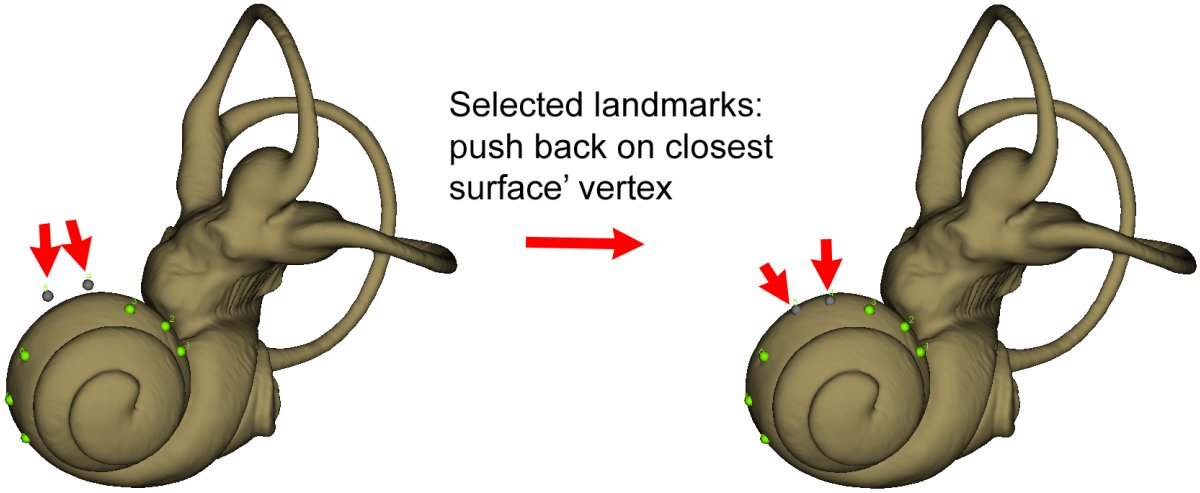


Figure 10.2: Push back selected landmarks on surface's closest vertex. A: two landmarks (4 and 5) laying outside the surface of the left inner of a *Galago moholi* specimen have been selected. B: these 2 landmarks now lay on the surface. This action can be undone/redone.

10.4 Selected landmarks: push back on closest surface's vertex

When set via pressing "L" + left click, landmarks are always positioned at a surface's vertex coordinates. Selected landmarks can be subsequently moved manually to other locations (for instance, if you want to place a given landmark in the middle of a canal or of a foramen, or between two unfused bones). However, you may sometimes want to push back automatically some selected landmarks to the position of the closest surface's vertex available. This can be achieved using this option (see for instance Fig. 10.2 p.91).

10.5 Selected landmarks: change orientation according to surface's normals

When set via pressing "L" + left surfaces, landmark orientation is that of the vertex on which it is placed. Selected landmarks' orientation can be subsequently moved manually. However, you may sometimes want to reset one or several landmarks' orientation automatically to that of the normal of the closest surface's vertex available. This can be achieved using this option (see for instance Fig. 10.3 p.92).

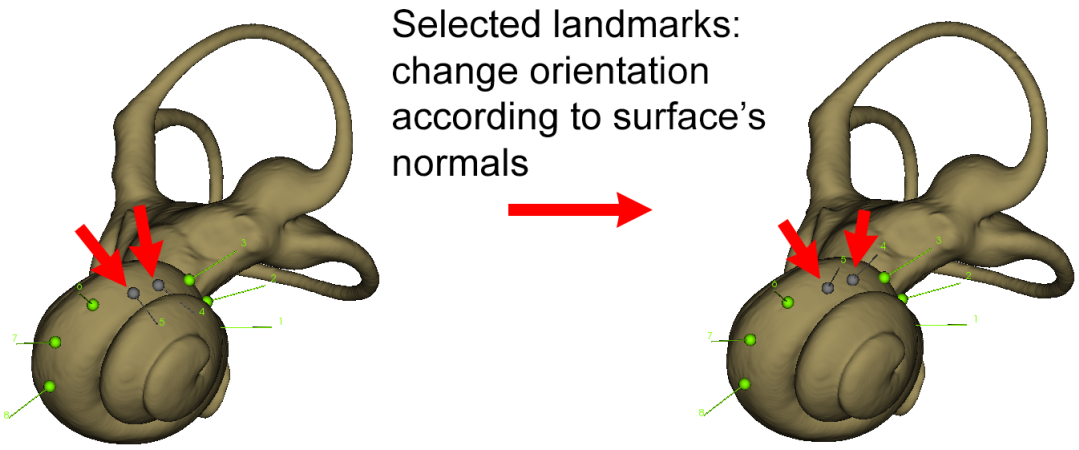


Figure 10.3: Change orientation according to surface's normals. A: two landmarks (4 and 5) the surface of the left inner of a *Galago moholi* specimen have been selected. B: these 2 landmarks' orientation now is the same as that of the closest vertices. This action can be undone/redone.

10.6 Selected curve nodes and curve handle landmarks

Before reading this paragraph, in order to get a good understanding about how curves are constructed in MorphoDig, please make sure you have read carefully the section 7.5 p.56.

10.6.1 Move curve handles semi-automatically

This option allows saving a lot of time when creating 3D Bezier curves with MorphoDig (see Fig. 10.5 p.93). Also see "working with curves" tutorial for further details regarding curve digitization with MorphoDig).

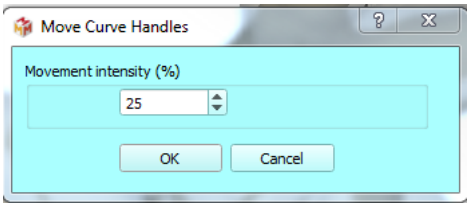


Figure 10.4: Mode handles window

Requirement : at least a handle landmark ("purple" landmark) must be selected. Depending on whether selected curve handles lie within the curve, at the start of the curve or at the end of the curve, their displacements differ (see Fig. 10.6 p. 94)

10.6.2 Normal nodes (red): change as starting modes (dark green)

This option makes it possible to decompose a "long" curve passing through a given number of curve nodes into distinct **"independent"** curve segments. A practical example is shown in

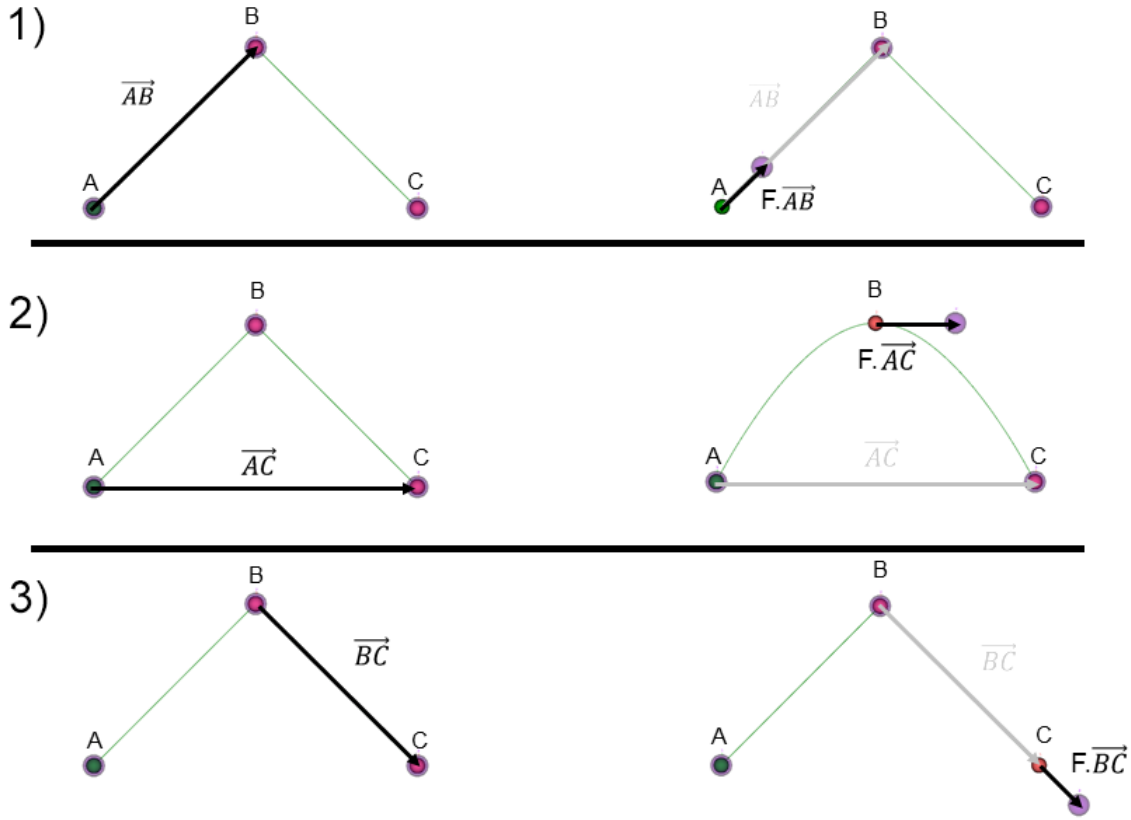


Figure 10.5: Moving handles semi-automatically.
1. Left: curve handle is associated to a curve starting point (A), and a following point (B) exists. Vector $\overrightarrow{AB}$ is computed, as well as its length |AB|. Right: curve handle associated to A is moved from point A along $\overrightarrow{AB}$. Displacement length=movement intensity/|AB|.
2. Left: curve handle is associated to a point B lying between two points (A and C). Vector $\overrightarrow{AC}$ is computed, as well as its length |AC|. Right: curve handle associated to B is moved from point B along $\overrightarrow{AC}$. Right: displacement length=movement intensity/|AC|.
3. Left: curve handle is associated to a curve ending point (C), and a preceding point (B) exists. Vector $\overrightarrow{BC}$ is computed, as well as its length |BC|. Right: curve handle associated to C is moved from point C along $\overrightarrow{BC}$. Displacement length=movement intensity/|BC|.

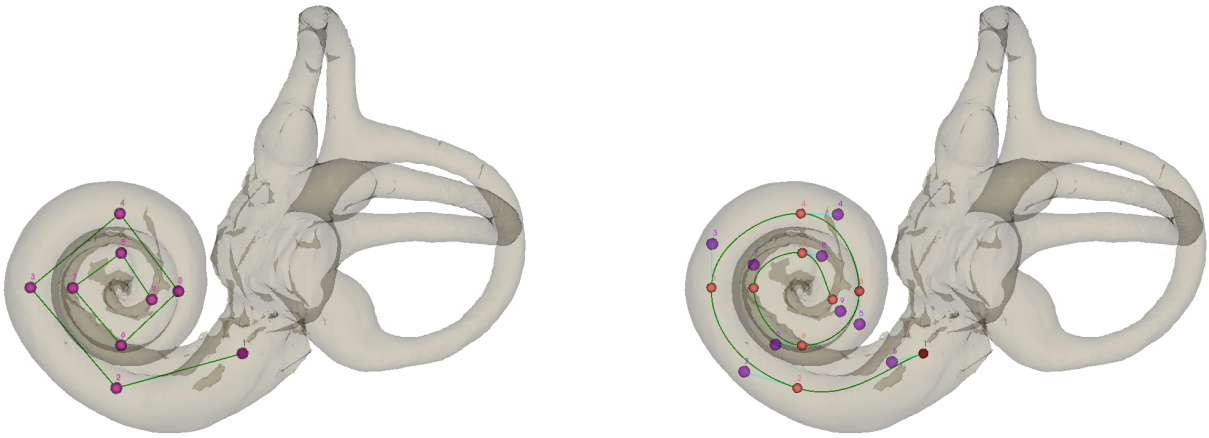


Figure 10.6: Example of curve handles semi-automatic displacement (movement intensity: 25%).

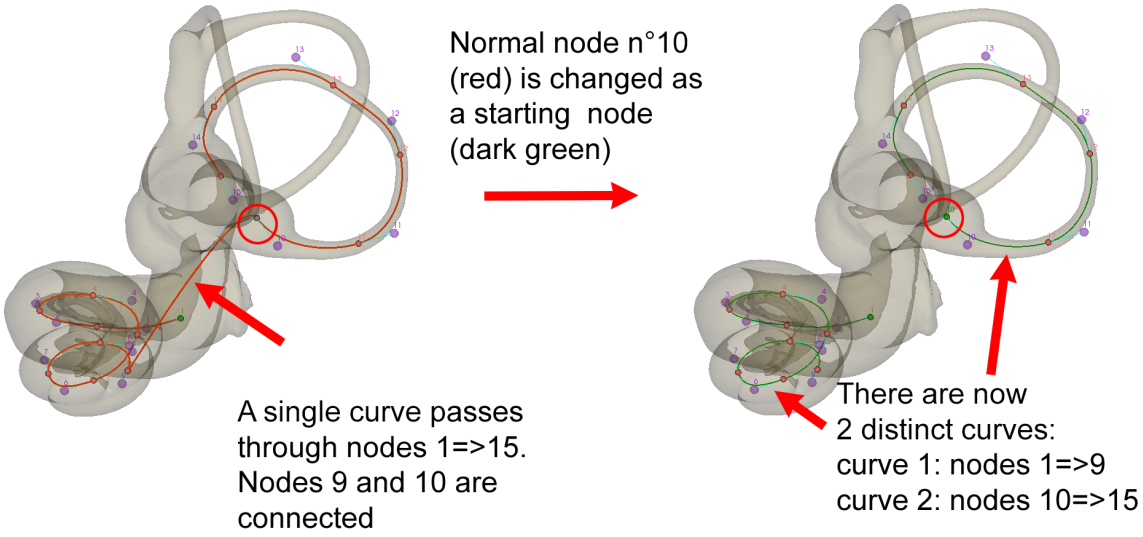


Figure 10.7: Curve decomposition into segments (left bony labyrinth of *Galago moholi*). Left: a single curve passes through 15 curve node landmarks. Right: curve node n°10 was defined as a new curve starting point. Now, there are two curves. The first curve passes through the cochlea only (nodes 1 to 9). The second one passes through the lateral semicircular canal (nodes 10 to 15).

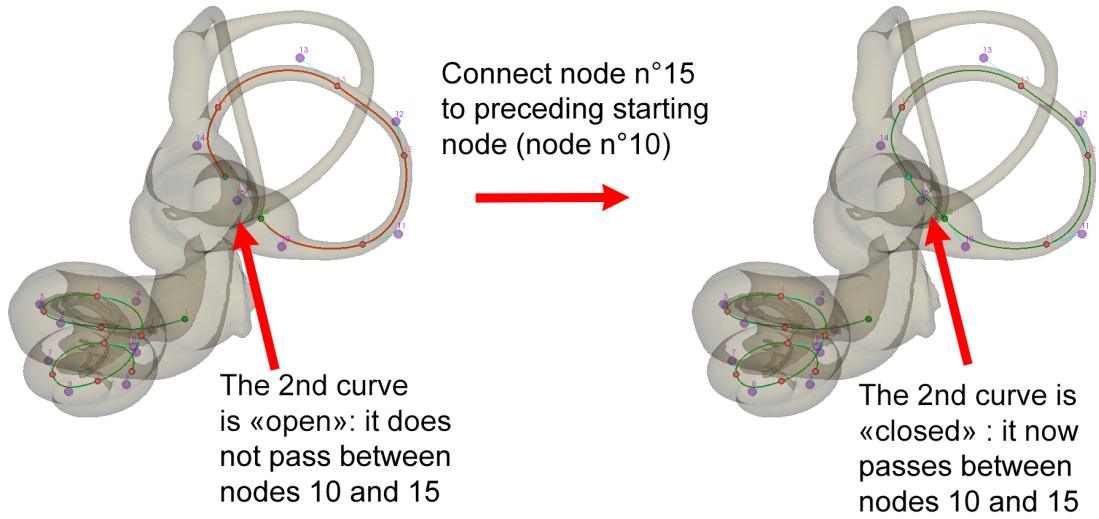


Figure 10.8: Closing curve segments (left bony labyrinth of *Galago moholi*). Left: the curve segment passing through the lateral semicircular canal is "open": the curve does not pass through the first curve node (node n°10) and the last one (node n°15). Right: curve node n°15 was connected to the preceding curve starting node n°10. Now, the curve passing through the lateral semicircular canal is "closed".

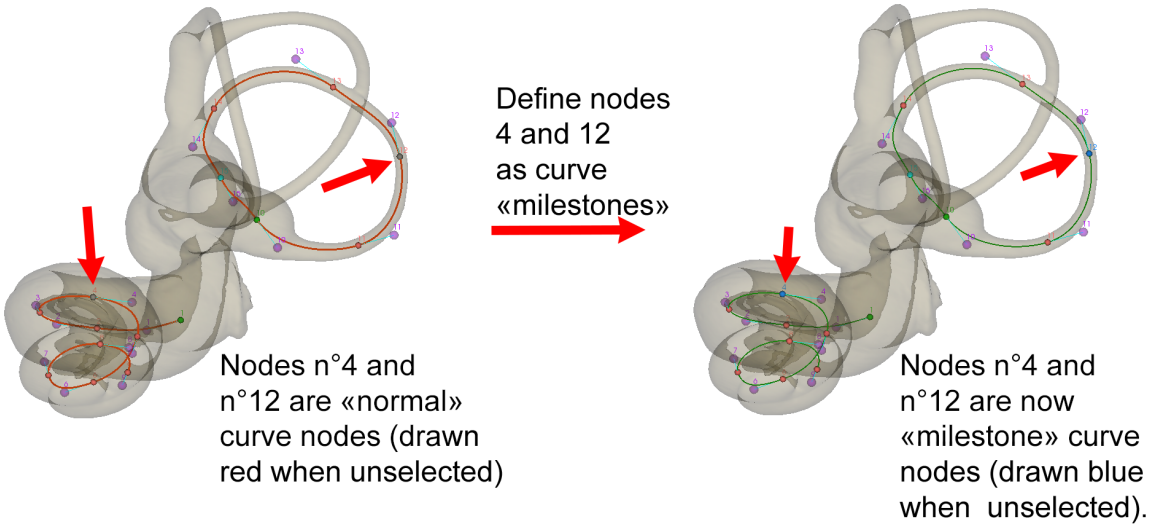


Figure 10.9: Curve decomposition into **contiguous** segments (left bony labyrinth of *Galago moholi*). Left: curve 1 (nodes 1-9) passes through the cochlea; curve 2 passes through the lateral semicircular canal (nodes 10-15). Right: curve nodes 4 and 12 were defined as curve milestones. When exporting curve infos or landmarks, MorphoDig now considers that 4 curve segments exist: curve 1 (nodes 1-4); curve 2 (nodes 4-9); curve 3 (nodes 10-12); curve 4 (nodes 12-15).

Fig. 10.7 p. 94. Selected node landmarks will be given flag “1". Please also refer to section 7.5 p.56.

10.6.3 Normal nodes (red): connect to preceding starting nodes (cyan)

This option makes it possible to "close" curve segments by connecting the last node of a curve segment to the starting point of that curve segment. A practical example is shown in Fig. 10.8 p. 95. Selected node landmarks will be given flag “3". Please also refer to section 7.5 p.56.

10.6.4 Normal nodes (red): define as milestone (blue)

This option makes it possible to decompose a curve passing through a given number of curve nodes into **"contiguous"** curve segments. A practical example is shown in Fig. 10.9 p. 95. Each curve milestone is involved into 2 curve segments: it acts as the ending node of a given curve, and as the starting point of the following curve. Curve milestones are given flag “2".

10.6.5 Reset selected nodes to Normal nodes (red)

Special nodes (curve starting points, nodes connected to their preceding starting points, and curve milestones) are reset back to a "normal" state when using this option. Selected landmark are given flag “0".
Further information regarding curve use in MorphoDig is available in the section “File → Curves" section (section 7.5 p.56) and in the tutorial “working with curves".

10.7 Edit color of all selected flag landmarks

Using this option, you can modify the color of all selected flag landmarks at once.

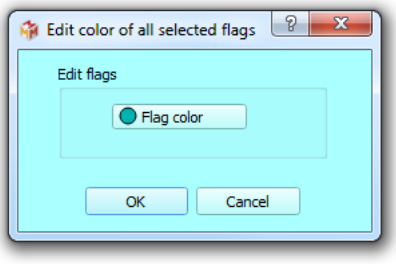


Figure 10.10: Edit color of all selected flags dialog

10.8 Edit length of all selected flag landmarks

Using this option, you can modify the length of all selected flag landmarks at once.

Figure 10.11: Edit length of all selected flags dialog

10.9 Update all selected flag landmarks' color to that of the closest vertex

This option will set the color of all selected flag landmarks to that of the closest vertex found in all opened surface objects. A practical example is shown in Fig. 10.12 p. 98.

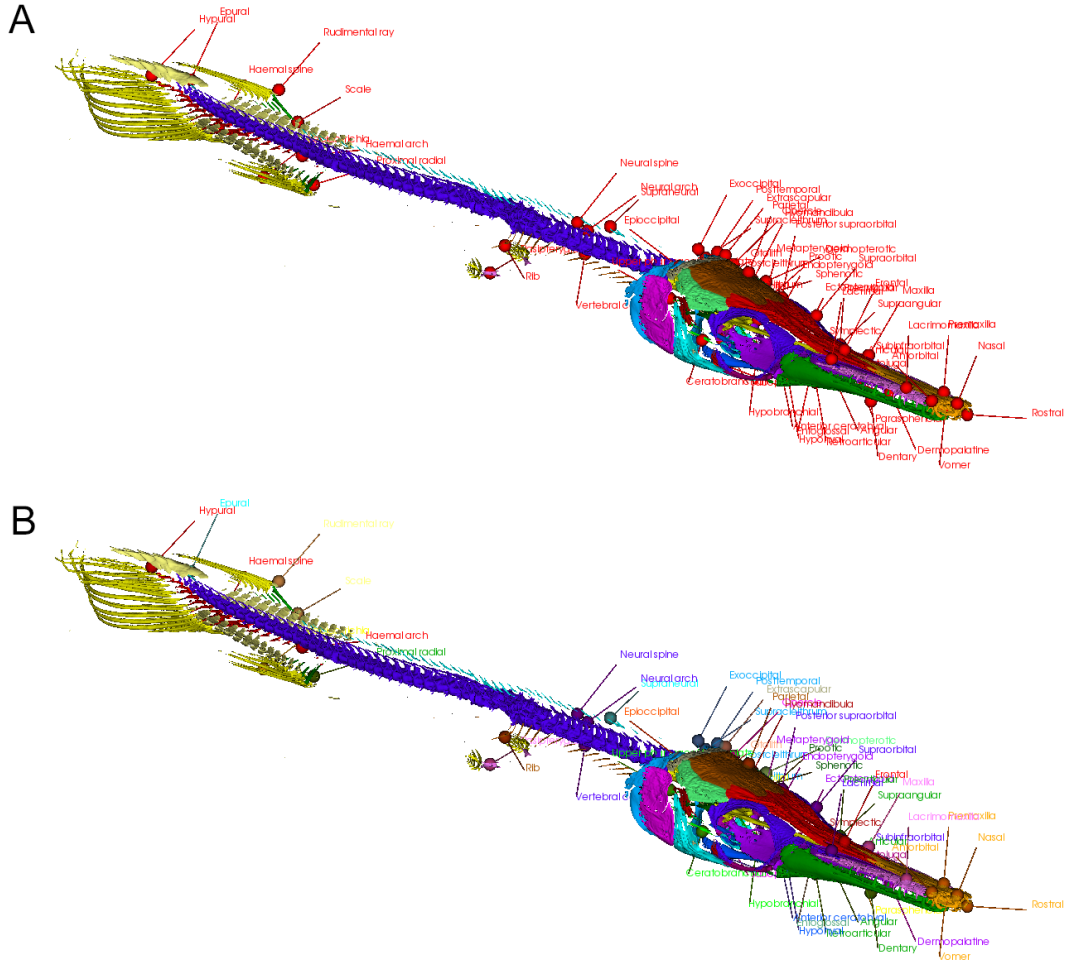
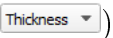


Figure 10.12: Setting the color for all flags according to the closest vertices' color of a tagged 3D model of skeletton of *Atractosteus tristoechus*. A: the flags are initially all drawn in red. B: all flags are now given colors that correspond to that of the closest vertices found for each flag.

Chapter 11

Menu Scalars

Contents

Simple numeric quantities (=numbers, or "scalar values") can be associated to each vertex of a given surface, and are referred to as "scalar arrays" in this section. As stated earlier, a given unselected surface can be colored using the currently active scalar array, if that surface contains that scalar array (see also Fig. 4.1-B p.20). To do so, the **array** display mode button must be pressed (), and a scalar array must be selected as the currently active array (ex: 

). The way scalar arrays are translated into colors can be set up using color maps, also referred to as "Lookup tables" (LUT) or color transfer functions.

11.1 Open scalars window

The "Scalars" window can be opened by clicking on "  " (see Fig. 11.1).
Available controls:
1: Active scalar array: choose among the available scalars the one which will be displayed.

2: Remove selected scalar array: deletes currently active scalar array from all opened surfaces, or from selected surfaces only. This option is useful if you plan to save surfaces in the .vtk format and do not want MorphoDig to save associated scalar values (this will save some disk space).

3: Active color map: choose among currently available colormaps. See Fig. 11.2 for a practical example.

4: Color map controls: Four buttons are available. a:reinitialize color map (only possible for the 2 first predefined color maps). b: export color maps(s). When clicked, the export color map(s) dialog shows up (see Fig. 11.3), in which you may define more precisely the color map(s) that should be exported. c: change active color map name (can be only done on custom color maps). d: delete active color map (only custom color maps can be deleted).

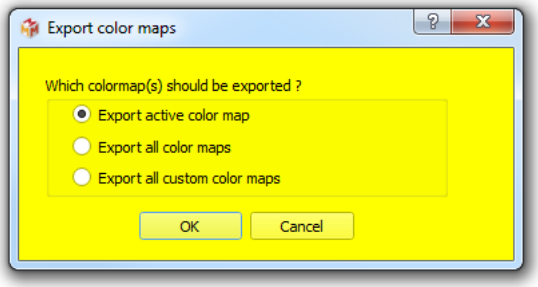


Figure 11.3: Export color maps dialog.

5: Opacity control points : opacity control points of the active color map. Such control points can be added (left click on a line), deleted (select one control point using the left mouse button then press "delete") and edited (drag one control point using the left mouse button) interactively.

6: Color control points : color control points of the active color map. Such control points can be added (left click on an empty zone), deleted (select one control point using the left mouse button then press "delete") and edited (drag one control point using the left mouse button) interactively.

7: additional color map controls. From top to bottom. a: set colormap range to match the global active scalar min and max found for all currently opened surfaces. b: reverse color color control points (see Fig. 11.4). c: reverse opacity control points (see Fig. 11.5). d: Save

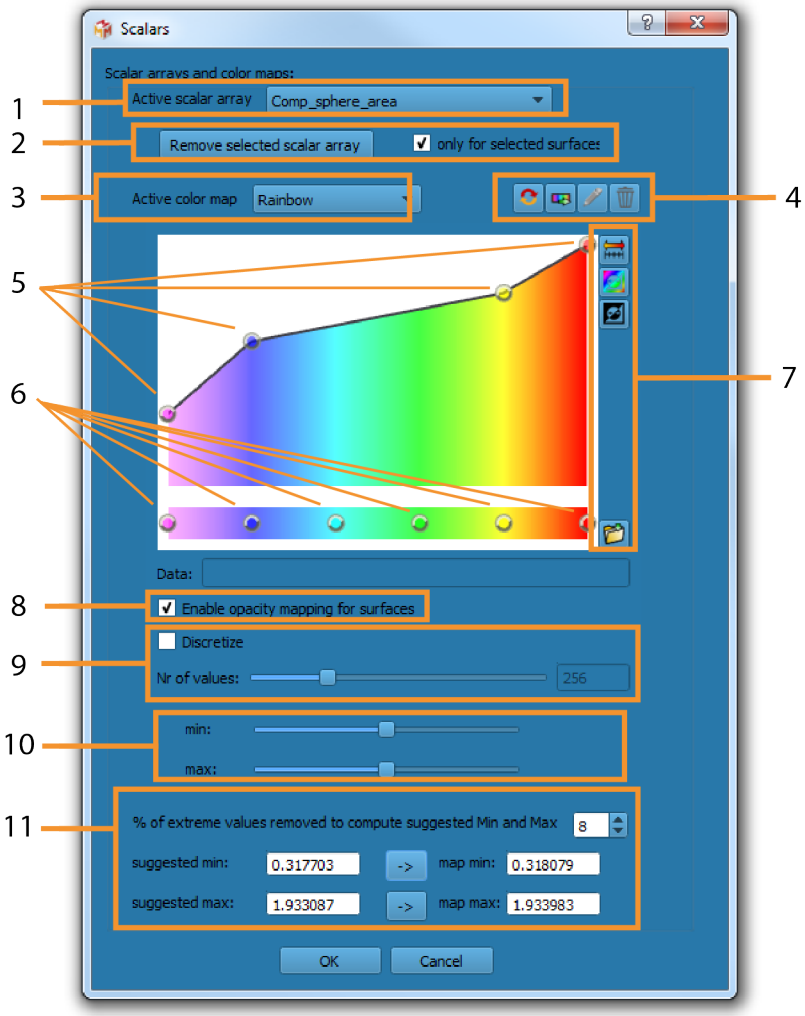


Figure 11.1: Scalar control window. This window is divided in different subsections. **1)** choose current 3D rendered scalar array. **2)** the active scalar array can be deleted from selected/all surface objects in this sections. **3)** choose current active color map, which transforms numbers into color and opacity on the screen. **4)** operations on the active colormap. a: reinitialize color map. b: export colors map(s) inside a .MAP file. c: change active color map name. d: delete active color map. **5)** opacity control points of the active color map. **6)** color control points of the active color map. **7)** modification controls of the active color map. a: set range to min and max. b: reverse color control points. c: reverse opacity control points. d: Save to preset = duplicate current active color map and create a new custom color map. **8)** enable/disable opacity mapping. **9)** discretize color levels, and choose number of levels. **10)** change min and max of color map. **11)** set min and max of color map based on suggested values. These suggested min and max values are computed in order to remove a percentage of most extreme minimal and maximal values found in the vertices of all opened surface for the active scalar.

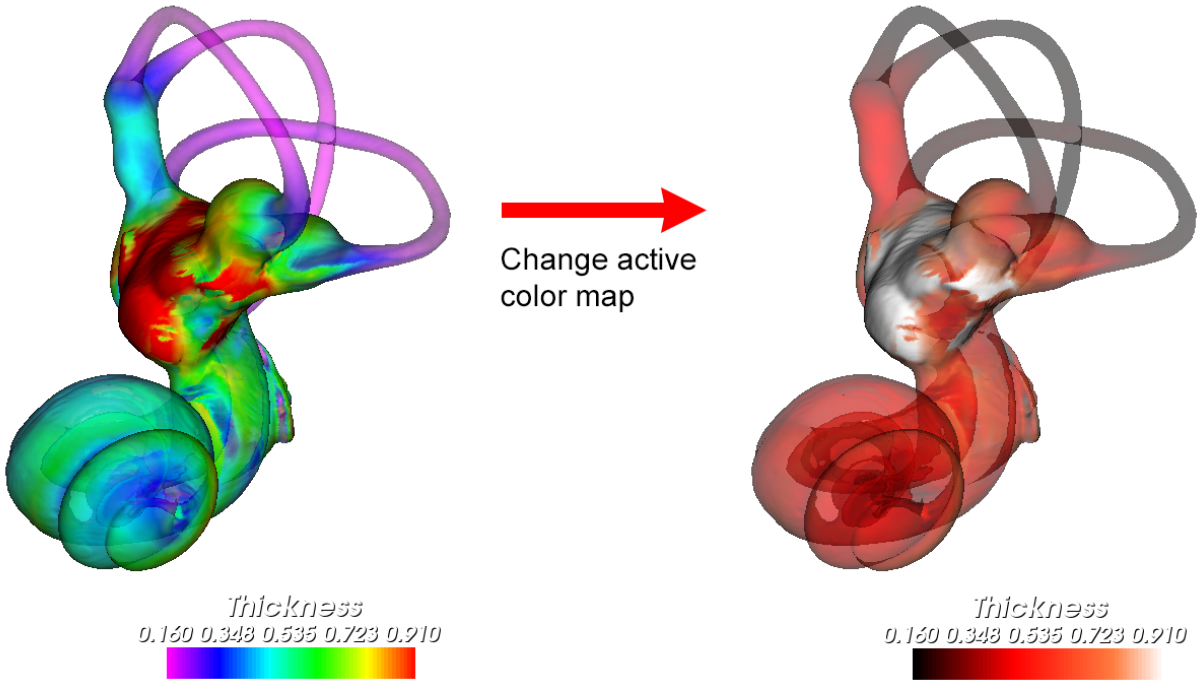


Figure 11.2: Example of active colormap modification. Left: left inner ear of *Galago moholi*. Active scalars: thickness. Colormap: rainbow. Right: the same inner ear after the active colormap has been set to "Black-red-white".

to preset = duplicate current active color map and create a new custom color map.duplicate current active color map and create a new custom color map.

8: enable/disable opacity mapping. When opacity mapping is disabled, all surface objects are rendered using their "global" transparency level.

9: discretize color levels, and choose number of levels. See Fig. 11.6 for a practical example.

10: change min and max of color map. You may use these sliders to change the minimal and maximal values of the active color map. By default, all colormaps range between 0 (min) and 1 (max), which may not be the most appropriate range to display a given scalar array.

11: set min and max of color map based on suggested values. As stated above, by default, all colormaps range between 0 (min) and 1 (max), which may not be the most appropriate range to display a given scalar array. You may set the minimal and maximal values of the active colormap based on suggested values, which are computed automatically based on the currently opened surface objects, and which are computed in order to use the color scale at its best. See Fig. 11.7 for a practical example.

11.2 Compute distance from camera for each selected surface

Computes distance (=depth) from camera for all vertices of all selected surfaces. This option may offer a better perception of the 3D structure of an object on a 2D screen representation. This option also makes it possible to compute an elevation map. See Fig. 11.8 for a practical example.

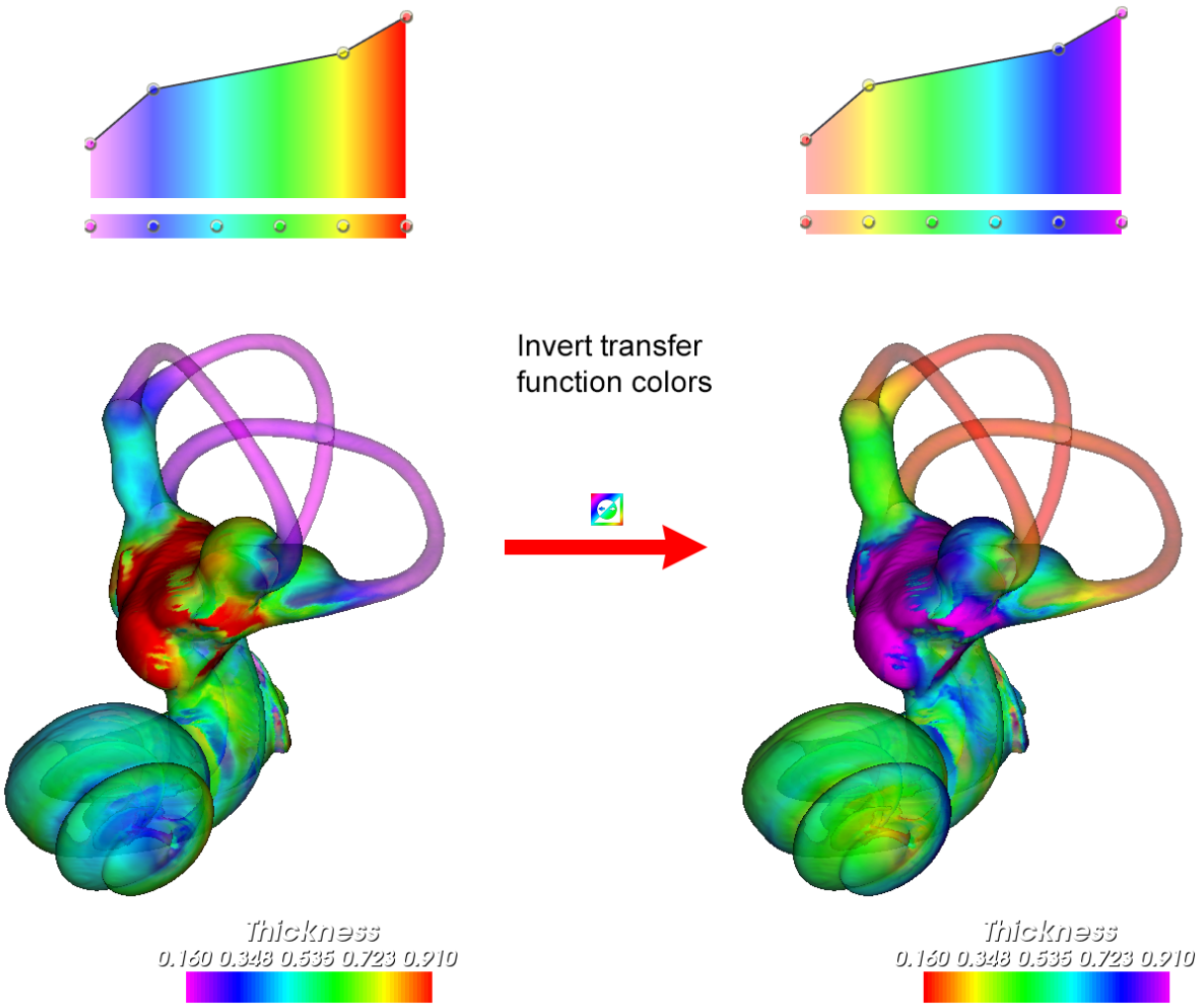


Figure 11.4: Example of color inversion of a colormap. Left: left inner ear of *Galago moholi*. Active scalars: thickness. Colormap: rainbow. Right: the same inner ear after the color control points of the rainbow colormap have been reversed.

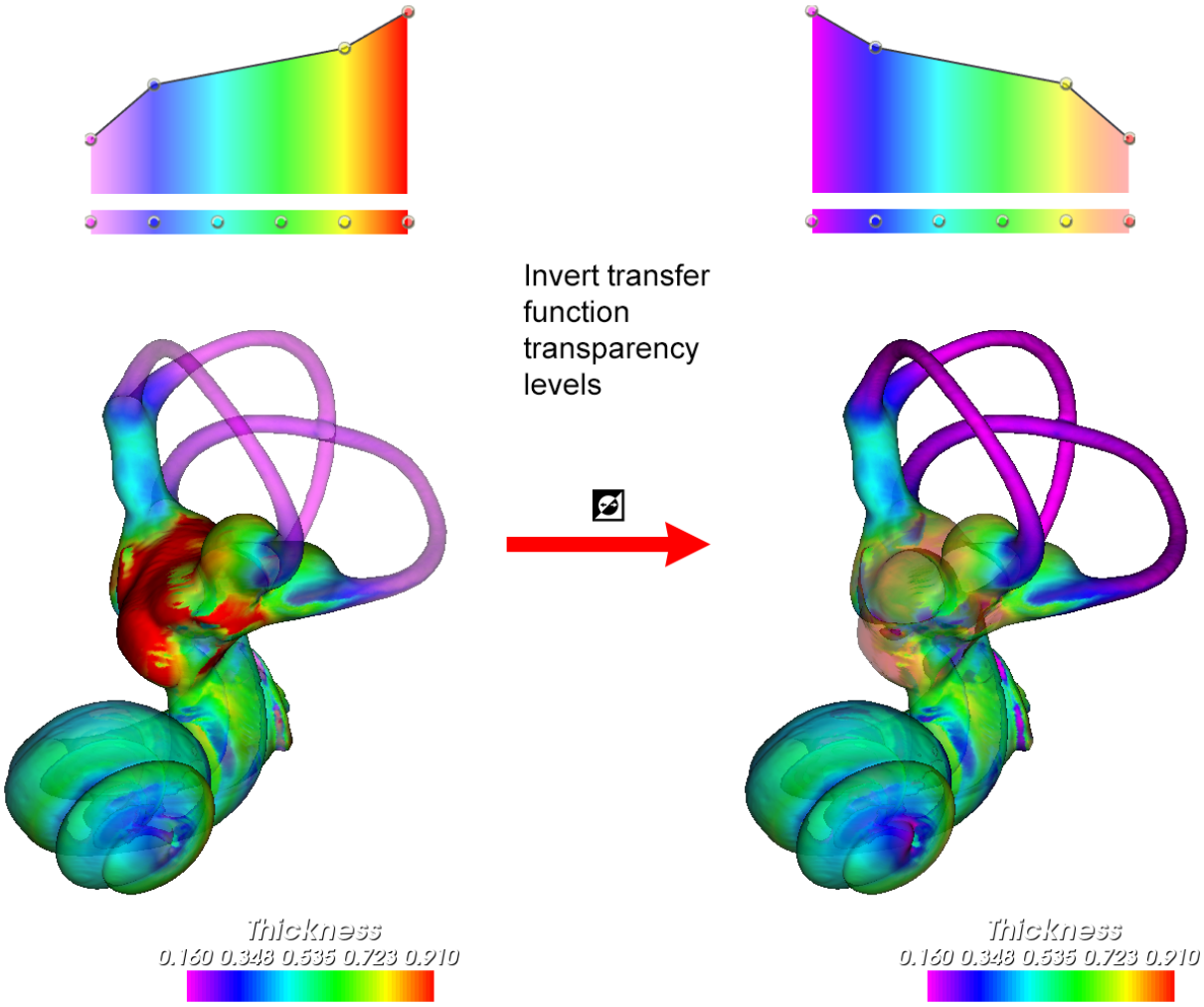


Figure 11.5: Example of opcacity inversion of a colormap. Left: left inner ear of *Galago moholi*. Active scalars: thickness. Colormap: rainbow. Right: the same inner ear after the opacity leveles of the rainbow colormap have been reversed.

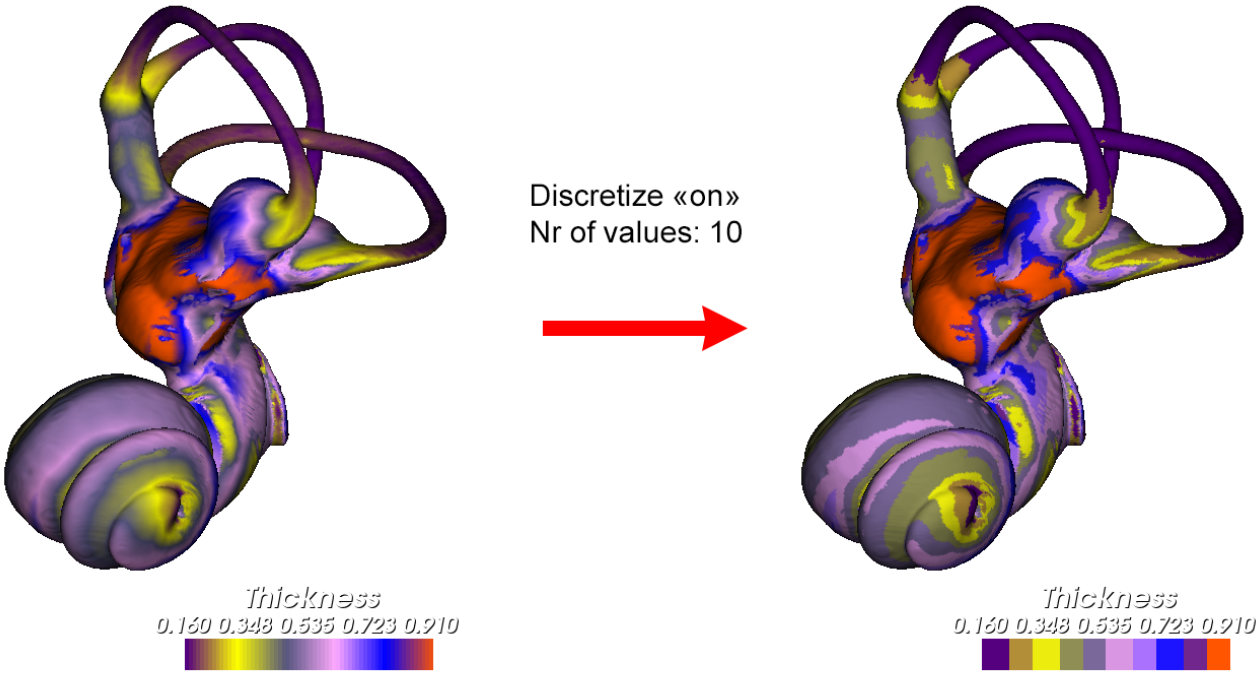


Figure 11.6: Example of colormap discretization. Left: left inner ear of *Galago moholi*. Active scalars: thickness. Colormap: custom. Right: the same inner ear after the "discretize" option has been turned on, and the nr of values set to 10.

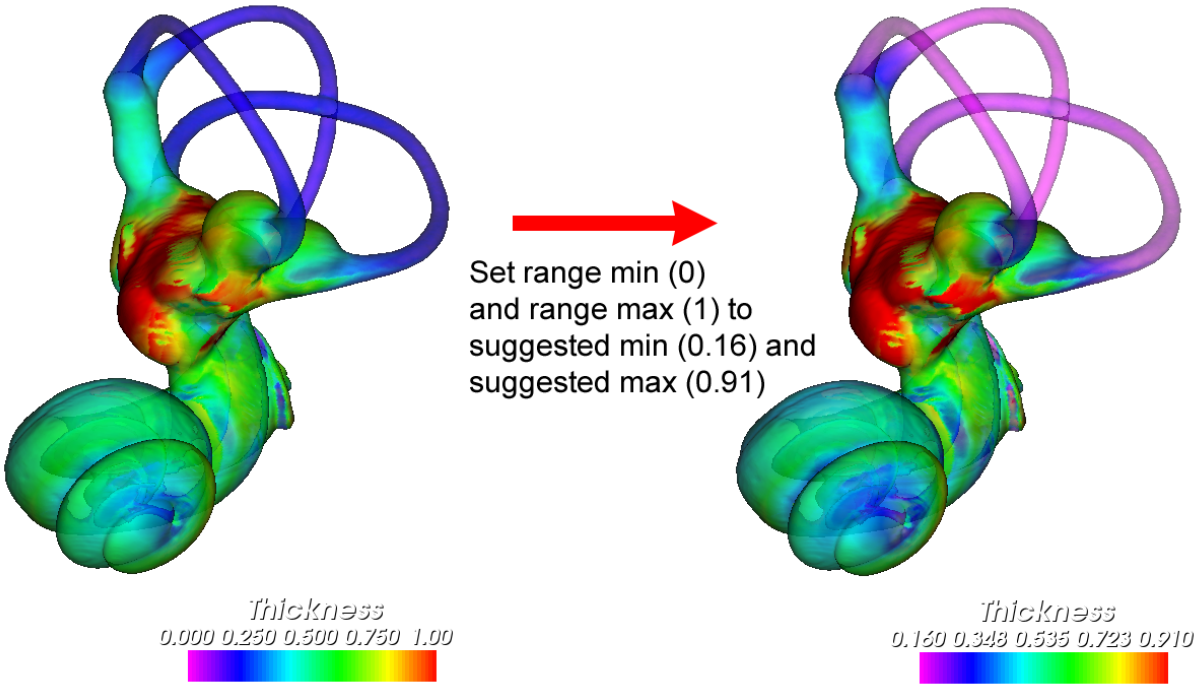


Figure 11.7: Example of suggested min and max usage. Left: Left inner ear of *Galago moholi*. Active scalars: thickness. Colormap: rainbow. Te rainbow colormap is set as by default to range between 0 (min) and 1 (max). Right: the same inner ear after the rainbow colormap range has been modified based on suggested min (0.16) and suggested max (0.91) values.

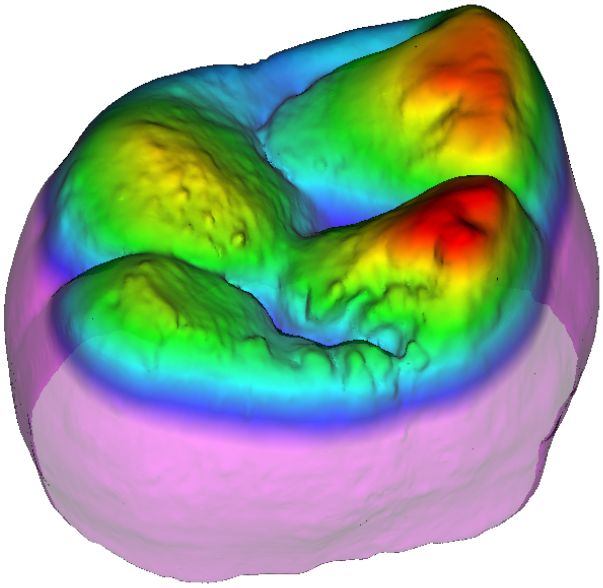


Figure 11.8: Example of camera distance scalar array. Displayed specimen: upper outer enamel surface of a second upper molar of *Homo sapiens*.

11.3 Compute thickness within each selected surface

Thickness within an object (see Fig. 11.9) is defined the following way: for a given vertex, the minimal distance between this vertex and other vertices in the direction opposite to that of the surface's normal. In order to minimize computation time, a maximal distance (Maximal thickness (size unit)) is asked to the user, in order to reduce the amount of vertices investigated at a given location. Also, in order to avoid to take into account only relevant vertices in that computation, an angular limit between investigated vertex normals (by default: 70°) is asked. Finally, a smoother scalar array output can be produced by setting the "thickness" value as the average of the distance found between a number of closest vertices found. See Fig. 11.10-A for a practical example.

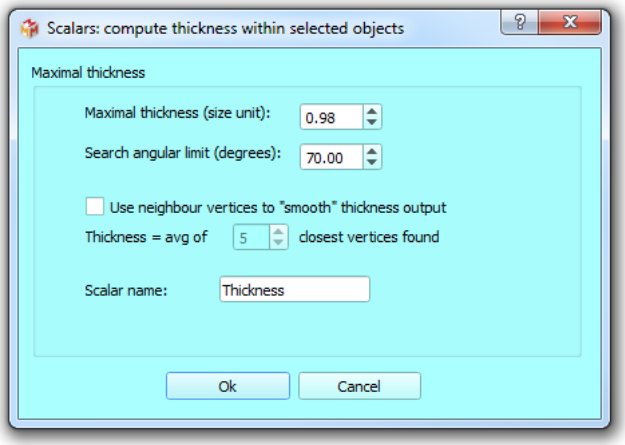


Figure 11.9: Thickness within a given surface.

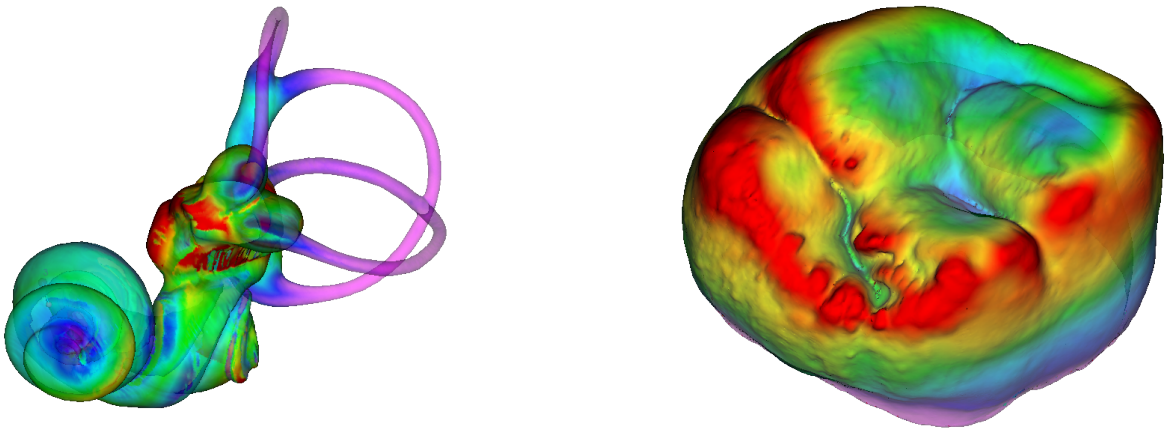


Figure 11.10: Examples of thickness computation. A: thickness within object. Specimen: left inner ear of *Galago moholi*. B: thickness between the outer enamel surface (OEJ) and the enamel-dentine junction (EDJ) of a second upper molar of *Homo sapiens*.

11.4 Compute thickness between two surfaces

Thickness between two objects is defined the following way (see Fig. 11.11): for a given vertex of the impacted object, the minimal distance between this vertex and other vertices of the observed surface in the same direction (if "invert normals of observed object" option is unchecked) or in the opposite direction (if "invert normals of observed object" option is checked) to that of the impacted surface's normal is computed. Again, in order to minimize computation time, a maximal distance (Maximal thickness (size unit)) is asked to the user, in order to reduce the amount of vertices investigated at a given location. Also, in order to avoid to take into account only relevant vertices in that computation, an angular limit between investigated vertex normals (by default: 70°) is asked. Finally, a smoother scalar array output can be produced by setting the "thickness" value as the average of the distance found between a number of closest vertices found. See Fig. 11.10-B for a practical example.

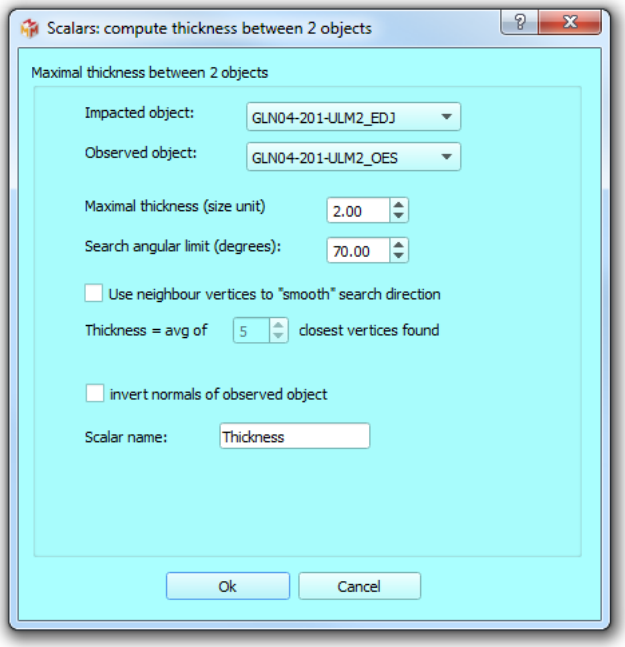


Figure 11.11: Thickness between 2 surfaces window

11.5 Compute distance between two surfaces

Vertex closest distance between two objects is computed as the minimal distance between all vertices vertex and other vertices of the observed surface, regardless of the direction. This option may be relevant if you want to compare the same object twice (for instance the object before and after some alteration) or want to compare two objects of very similar shape (for instance two inner ears belonging to two specimens of the same species). Before computing a distance map, it is strongly advised, in a preliminary step, to align the two compared surfaces (see section 9.3 p.86). The computed distance can be somewhat "smoothed" by being defined as the average value of a given number of closest vertices. Again, in order to minimize computation time, a maximal distance (in size unit) is asked to the user, in order to reduce the amount of vertices investigated at a given location. See Fig. 11.13 for a practical example.

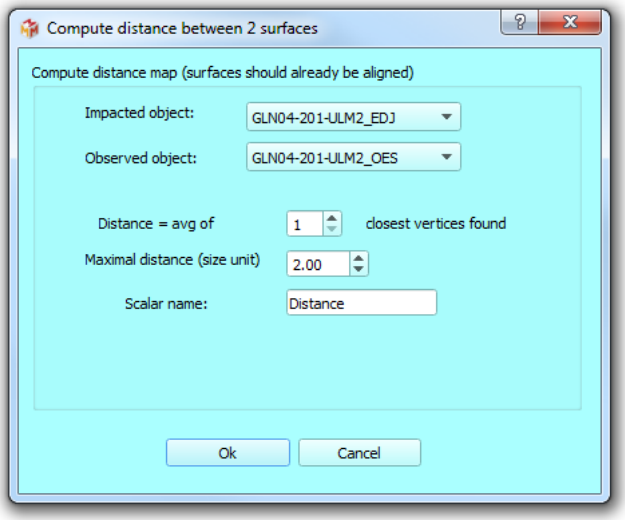


Figure 11.12: Distance between 2 surfaces window

11.6 Compute curvature for each selected surface

Curvatures are computed using the vtkCurvatures filter.
vtkCurvatures filter offers 4 ways to compute surface's curvature at each vertex (see Fig. 11.15):
- Principal maximal curvature
- Principal minimal curvature
- Gaussian curvature
- Mean curvature.

See vtkCurvatures' documentation for further details.

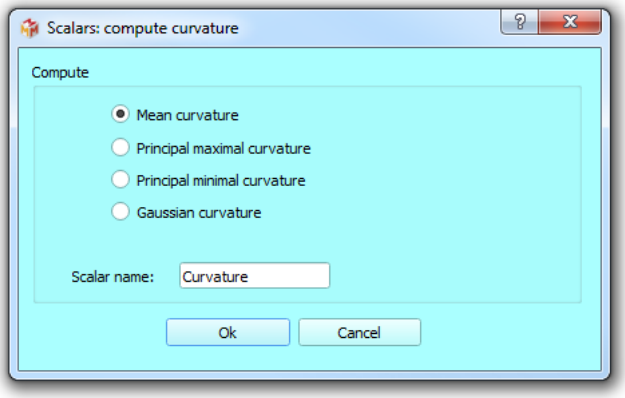


Figure 11.14: Curvature dialog

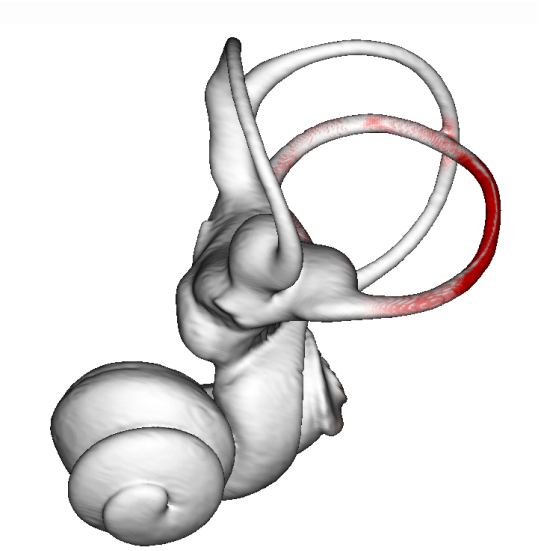


Figure 11.13: Example of distance map between two surfaces. An inner ear of *Galago moholi* was virtually distorted in the lateral canal region. The corresponding representation of the deformation is shown here using a distance map: in red can we identify the most distorted regions.

11.7 Compute complexity for each selected surface

In section 7.10.2 p.66 and section 7.10.3 p.66, proxies for shape complexity (= how much area can be enclosed within a given volume) are described. The limit of those approaches is that they produce single values for a whole surface, and do not take into account shape variation within surfaces. However, a single biological object can express a lot of shape variability. In this section, we extend the approach of shape complexity described earlier: we provide surface complexity measurements for all vertices of a surface. To do so, for each vertex, a local surrounding area is extracted and shape complexity is measured for this local area. The consequence of this approach is that these measurements depend on the extent of the local area investigated for each vertex. For a given surface, the "local" area investigated is a sphere of radius "R". By default, for a given surface, R is computed as follows:

$$R = \frac{Avg(length(PC1) + length(PC2) + length(PC3)))}{18} \tag{11.1}$$

The numerator of this equation is in our view a sensible estimate of the "average extension" of an object in 3D space. It is computed as is the average of the lengths of a given object along its 3 principal extension axes. This measurement is obtained by performing a principal component analysis on the vertex coordinates of a given surface. The denominator (18) is set empirically, as it seems to provide a reasonable estimate of "locality" for all vertex locations,

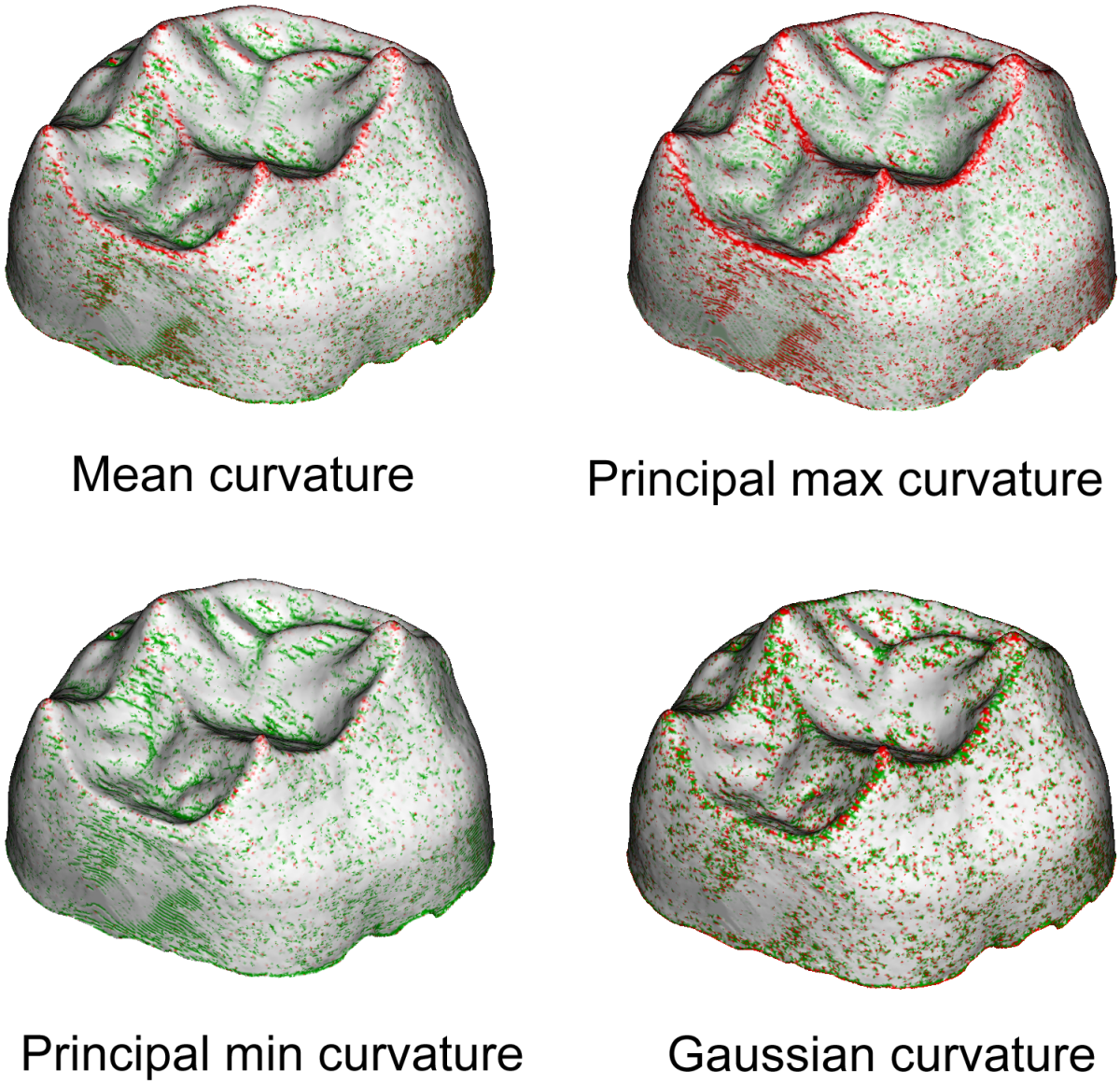


Figure 11.15: Examples of 3D rendering of “Curvature" scalars. Scalar mode is active, the rainbow color scale is used. Specimen: enamel dentine junction (EDJ) of the second superior molar of a medieval human from Sains-en-Gohelle (France). Image credit: Mona Le Luyer.

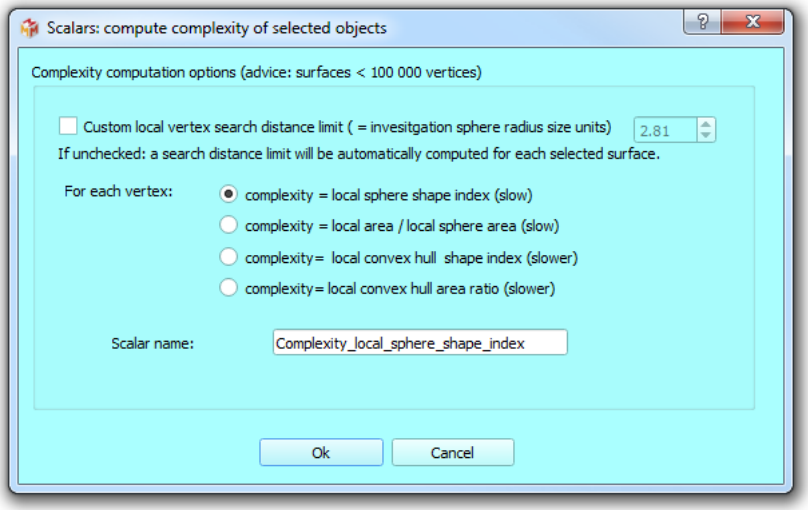


Figure 11.16: Complexity dialog.

regardless the global shape of the investigated surfaces. However, R can be set to a custom value in the Scalars complexity dialog (see Fig. 11.16).

Four complexity measurements are provided:

11.7.1 Local Convex hull Area Ratio (LCh-AR)

For each vertex V, the surrounding surface's local region enclosed within an area defined by a local sphere of center V and radius R is extracted. Then the local convex hull normalized area ratio (ChAR) is computed for this local region. ChAR is defined as the ratio between the extracted local surface area and the surface area of the convex hull computed for the same extracted local surface.

This measurement of surface complexity takes more time to compute (computation of a Convex hull for each vertex) than the LS-AR and LS-NSI indices (see below) based on the local sphere, but it has two advantages (see Fig. 11.17-B p.11.17):

- very low "border" artifacts (complex structures located at the outskirts of surfaces are given high complexity values).
- contrary to the LCh-NSI index (see below), isolated very flat (and non-complex) structures are given, as they should be, low complexity values .

11.7.2 Local Sphere Area Ratio (LS-AR)

For each vertex V, the surrounding surface's local region enclosed within an area defined by a local sphere of center V and radius R is extracted. Then complexity is computed as the ratio between the area of the extracted surface and the area of the sphere of radius R.

This measurement of surface complexity takes less time to compute than the LCh-AR index, but it has one main drawback (see Fig. 11.17-C p.11.17):
- border artifacts: as the denominator used to construct this index is based on the surface area of the local sphere for a given vertex V, vertices located at the outskirts of a surface are given underestimated complexity values (and as a general rule, LS-AR tend to underestimate complexity in regions containing a small number of vertices) .

11.7.3 Local Convex hull Normalized Shape Index (LCh-NSI)

For each vertex V, the surrounding surface's local region enclosed within an area defined by a sphere of center V and radius R is extracted. Then the local convex hull normalized shape index (ChNSI) is computed for this local region (see section 7.10.3 p.66 for explanations of ChNSI index).
This measurement of surface complexity takes more time to compute (computation of a Convex hull for each vertex) than the LS-AR and LS-NSI indices (see above and below) based on the local sphere, but it presents one advantage over the LS-NSI: (see Fig. 11.17-D) p.11.17:
- very low "border" artifacts (complex structures located at the outskirts of surfaces are given high complexity values).
However, this index has a considerable drawback: it produces overestimated complexity values for all very flat + very isolated areas (which are typically non complex areas). The reason is that the denominator used to construct this index is based on the surface volume of the local convex hull, which tends to be almost null for very flat and isolated areas. So if your structure presents such flat-and-isolated regions, we advise you not to use this index.

11.7.4 Local Sphere Normalized Shape Index (LS-NSI)

For each vertex V, the surrounding surface's local region enclosed within an area defined by a sphere of center V and radius R is extracted. Then the local normalized shape index (NSI) is computed for this local region (see section 7.10.2 p.66 for explanations of NSI index). This measurement of surface complexity takes less time to compute than the LCh-NSI and LCh-AR indices. Furthermore, it does not overestimate complexity in isolated flat regions, as the Ch-NSI index does (see Fig. 11.17-E p.11.17). However, in a similar way as the LS-AR index, as the denominator used to construct this index is based on the surface area of the local sphere for a given vertex V, vertices located at the outskirts of a surface are given underestimated complexity values (more border artifacts than the LCh-NSI and LCh-AR indices).

A practical example of surface complexity is found in Fig. 11.17 p.11.17. As a general rule, when possible, please consider to use the LCh-AR index as it is may produce less artifacts than the other 3 indices.

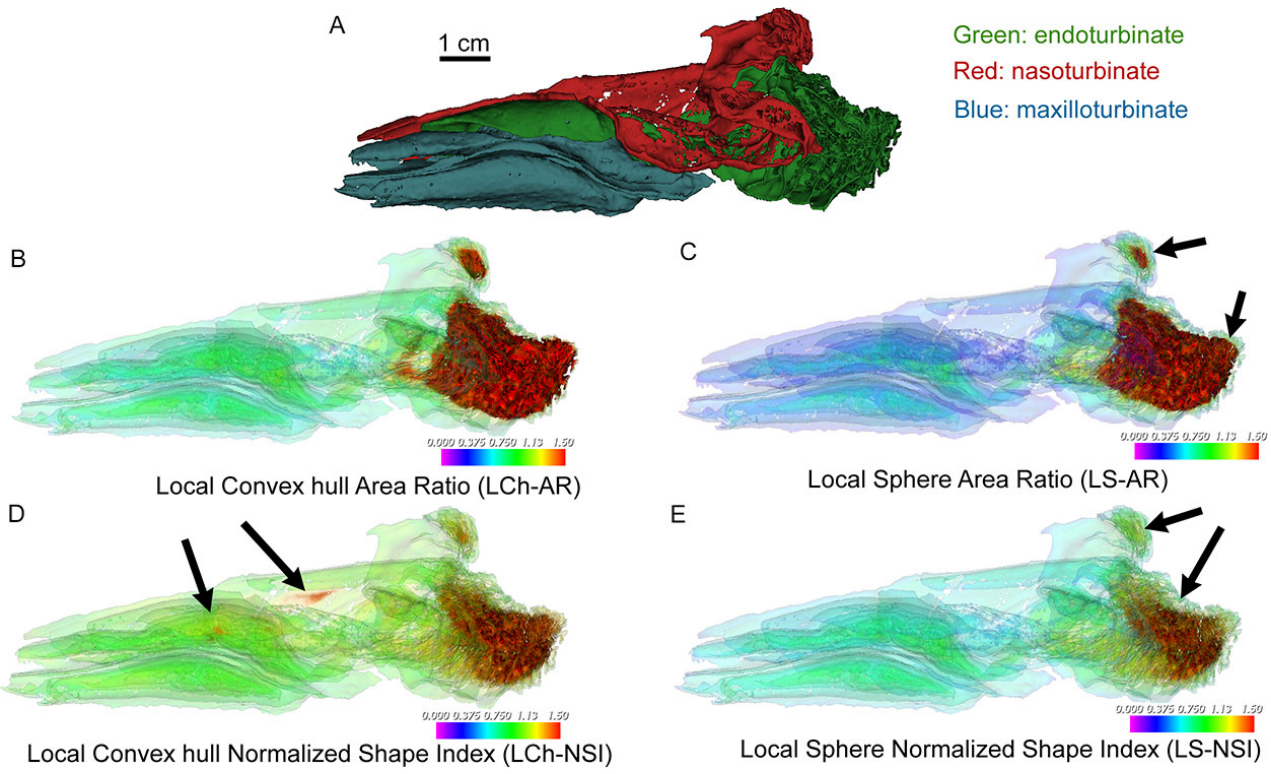


Figure 11.17: Local complexity scalar computed in turbinate bones of *Orycteropus afer*. **A**. Left endo, naso and maxillo turbinate bones viewed from the left side. Left-to-right: antero-posterior direction. Bottom-up: dorso-ventral direction. **B**: the Local Convex hull Area Ratio (LCh-AR) presents two advantages: no "border" artifacts, and no overestimation of complexity in flat-isolated regions. **C**: the Local Sphere Area Ratio (LS-AR) tends to underestimate complexity at the outskirts of complex structures ("border artifacts", see black arrows). **D**: the Local Convex hull Normalized Shape Index (LCh-NSI) tends to overestimate complexity is some very flat and isolated areas. See for instance the red areas in the center of the nasoturbinate and in the anterior part of the endoturbinate (black arrows). E: Local Sphere Normalized Shape Index (LS-NSI) presents no overestimation of complexity in flat-isolated regions, but tends to underestimate complexity at the outskirts of complex structures ("border artifacts", see black arrows). Image courtesy: Lionel Hautier and Mark Wright.

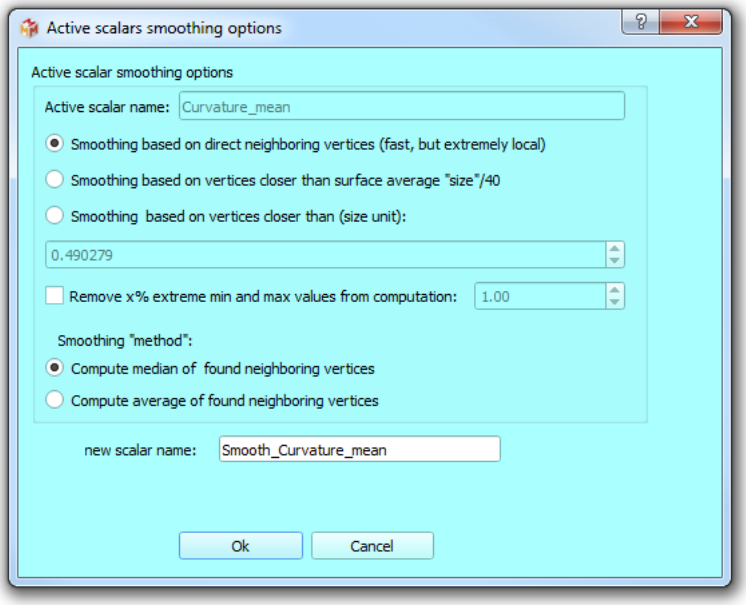


Figure 11.18: Scalar smoothing dialog. Smoothing can be computed using only direct neighbor vertices. A much larger area can also be investigated to produce a "smoothed" output. A percentage of extreme minimal and maximal values found can be excluded from the smoothing process. Also, the "smoothed" output can be computed as the average or as the median of the scalar values found for the neighbor vertices.

11.8 Smooth active scalars for each selected surface

It may happen that scalar values contain a lot of noise (see for instance the noisy results of the curvature computation in Fig. 11.15). In order to reduce noise and retrieve more biologically relevant information, scalars can be “smoothed" in different ways (see 11.18). Be aware that the obtained results depend very much on the size of the investigated local area for each vertex (based on direct neighbor vertices or on a much larger area), and on whether the new scalar value is computed as the average or as the median of the scalar values found for neighbor vertices (see Fig. 11.19).

11.9 Normalize or rescale active scalars for each selected surface

It may be useful to normalize scalar values between 0 and 1, or to rescale them for different purposes. For instance, scalar normalization makes it possible to compare scalar arrays of

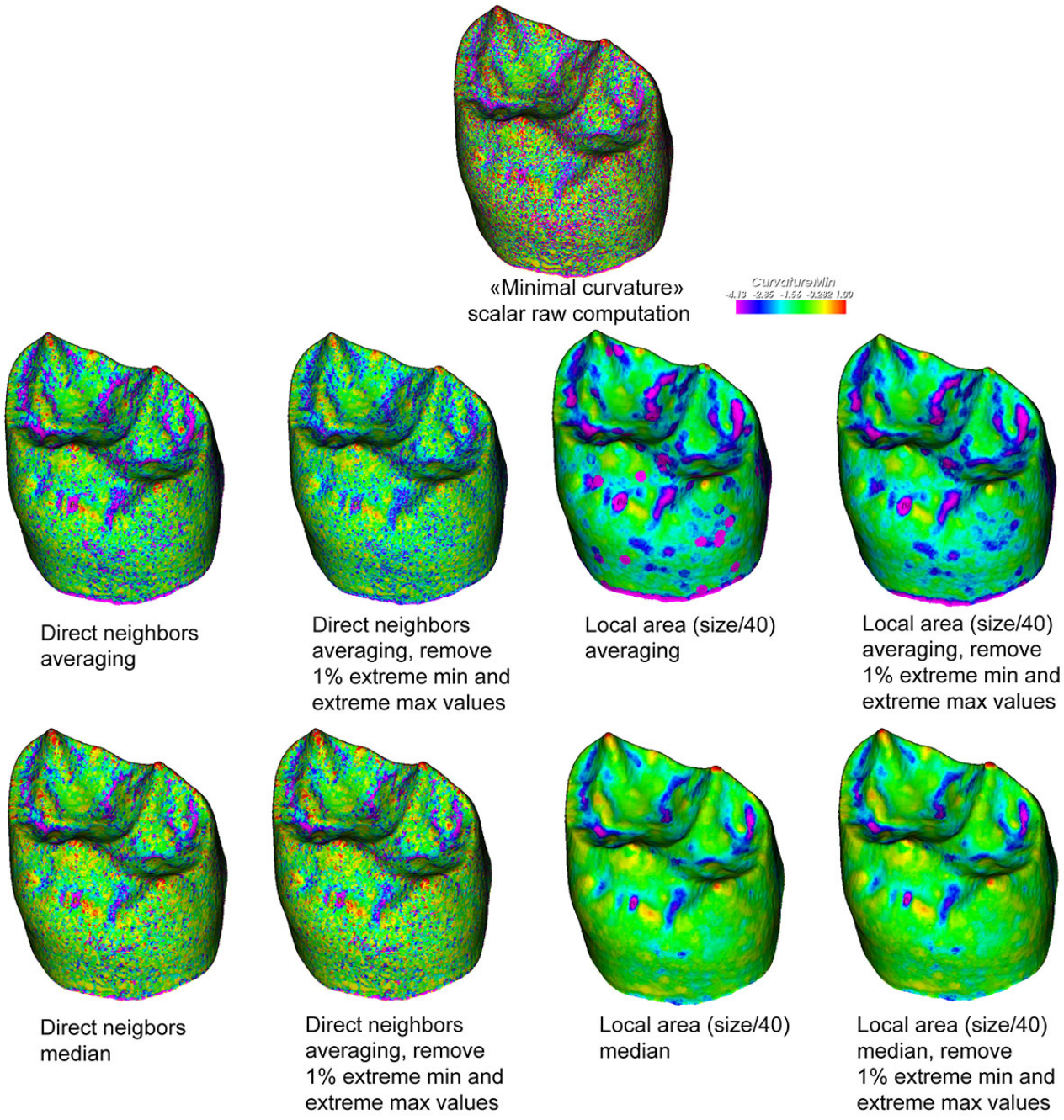


Figure 11.19: Smoothing scalars. Examples of 3D rendering of "Min Curvature" scalars. **Top** : "raw" minimal curvature, which is extremely variable locally, and hard to interpret. **Middle line**: averaging direct neighbors or averaging scalar values on a much larger area give different results. See also how much averaging scalar values is sensitive with extreme min and max values. **Bottom line** : computing a median based upon direct neighbors or upon scalar values found within a much larger area also give different results. The median smoothing "method"' is much less sensitive with extreme values (no difference seen when only 1% of the most extreme min and max values are removed). Specimen: enamel dentine junction (EDJ) of the second superior molar of a medieval human from Sains-en-Gohelle (France). Image credit: Mona Le Luyer.

different nature (and having very different min-max ranges) using a unique color scale ranging between 0 and 1. Also, in order to be able compare scalars such as bone thickness or enamel thickness variation between specimens of very different size, scalar normalization is a necessary step. The scalar normalization/rescale dialog is shown in Fig. 11.20 p.119. An example of normalization of different scalars within a single surface is shown in Fig. 11.21 p.120. An example of normalization of a single scalar in order to compare different specimens is shown in Fig. 11.22 p.121.

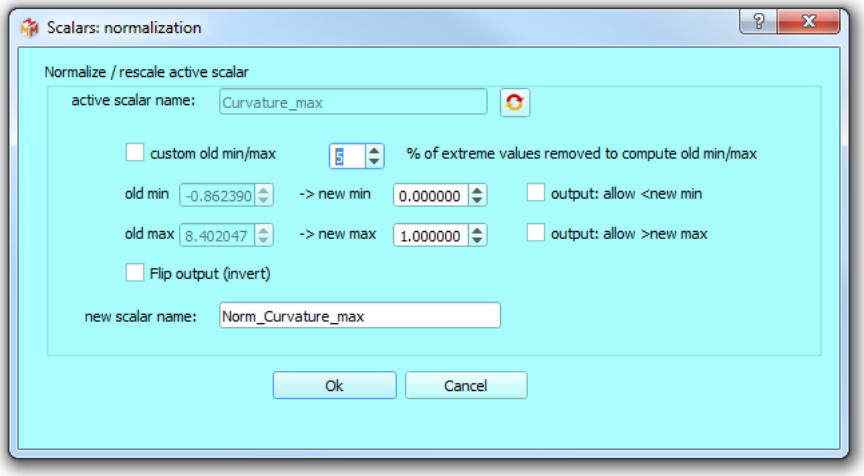


Figure 11.20: Scalar normalization/rescale dialog. The "old min" and "old max" value represent by default the minimal and maximal value of the active scalars within **all** currently opened surfaces. After the scalar normalization/rescale has been performed, "old min" will have been set to "new min", "old max" will have been set to "new max", and scalar values in between "old min" and "old max" will range between "new min" and "new max" using a linear interpolation. "old min" and "old max" can be given other values than the real min and max of the active scalar via two options. **1)** a given percentage of extreme min and max values can be removed from the current active scalar values (sometimes only a few extreme "strange" values can give a wrong idea of the "real" extant of the range of a given scalar). **2)** if the "custom old min/max" option is checked, old min and old max values can be set to any value. When one of these two options is used, the interpolated values produced may fall above "new max" or below "new min". Then you may decide whether you allow such "out of the range" values using the "allow $<$ new min" and "allow $>$ new max" options. If unchecked, interpolated values falling below "new min" will be set to "new min", and interpolated values falling above "new max" will be set to "new max". The "flip output" option will reverse the range of interpolated values between "new min" and "new max".

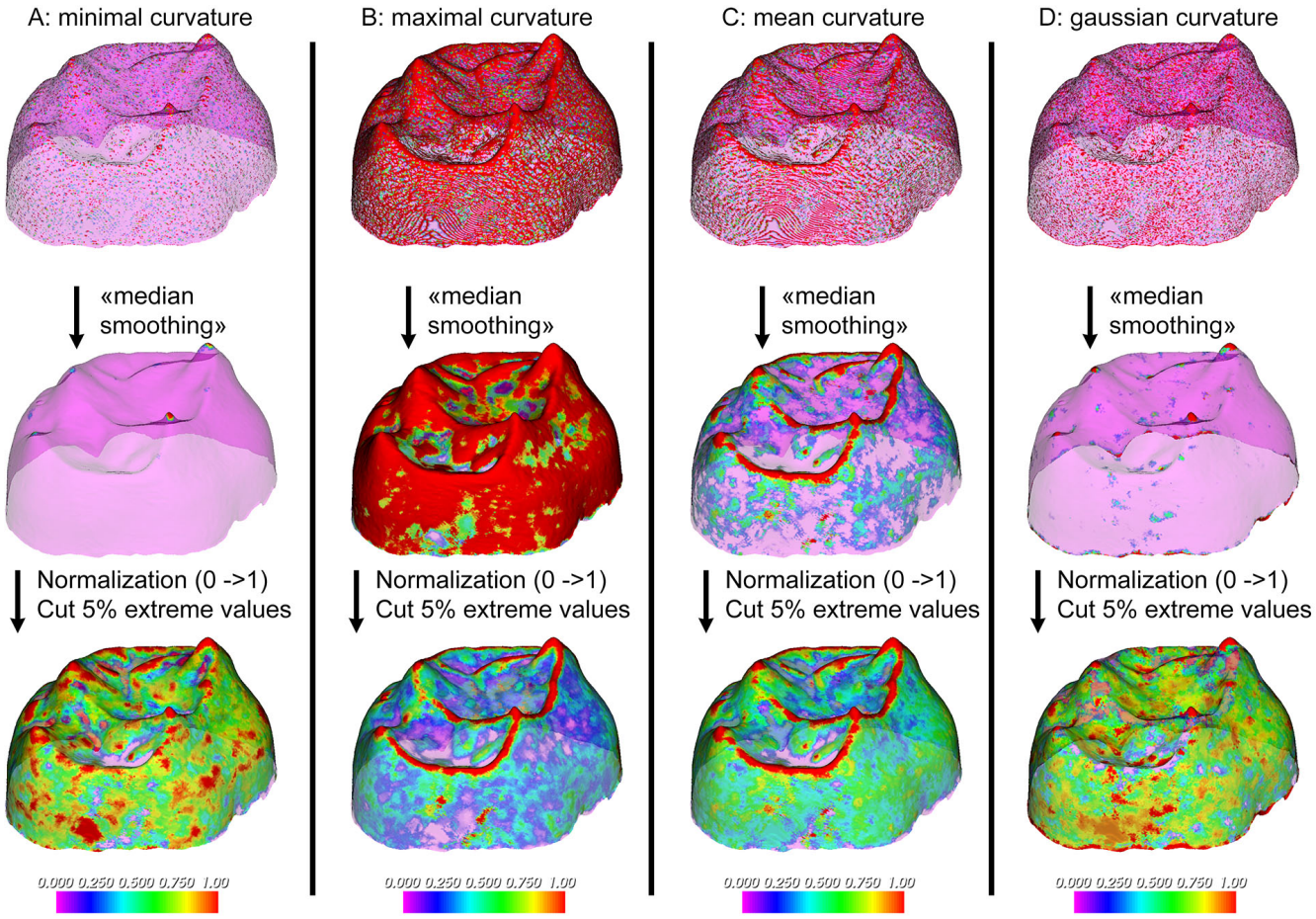


Figure 11.21: Normalization of different scalars within the same specimen. All illustrations use the same color table, ranging between 0 and 1. **A** : "Minimal curvature". **B** : "Maximal curvature". **C** : "Mean curvature". **D** : "Gaussian curvature". Top line: "raw" 4 scalars, which all express a lot of noise. Middle line: the 4 same "smoothed"' scalars after a median filter has been used. Bottom line: the 4 same scalars after normalization. The 5% most extreme minimal values and maximal values have been excluded from this normalization process, because they are clearly aberrant (most values range between -10 and 10 for all 4 raw scalars, but a few outliers have values above 10000 and below -10000). The visual output of the bottom line is easier to interpret. Specimen: enamel dentine junction (EDJ) of the second superior molar of a medieval human from Sains-en-Gohelle (France). Image credit: Mona Le Luyer.

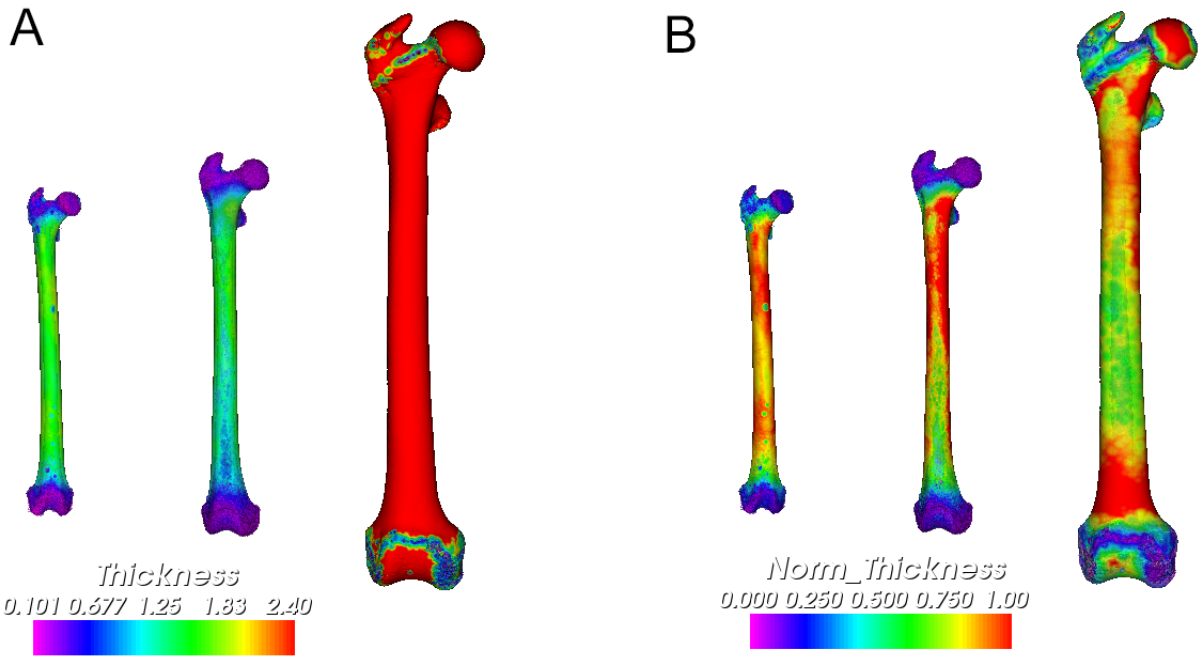
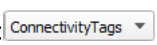


Figure 11.22: Normalization of a single scalar array in different specimens. From left to right in A and B: right femurs of *Cercopithecus mona*, *Erythrocebus patas*, *Papio* sp. **A**: "raw" bone thickness. See how bone thickness is much higher, generally speaking, for the femur of *Papio sp.*, a relatively much larger primate than in the two others. This makes it hard to identify and compare variation of bone thickness for these three specimens. **B**: normalized bone thickness. Direct description and comparison between these three specimens is now possible.

Chapter 12

Menu Tags

Contents

All vertices of different biological structures can be given a specific integer values (0, 1, 2, 3 ...), in order to identify them. Such integer arrays are referred to as "tag arrays". As stated earlier, a given unselected surface can be colored using the currently active tag array (identified by its "name"), if that surface contains a tag array of that name (see also Fig. 4.1-B p.20). To do so, the array display mode button must be pressed (), and a **tag** array must be selected as the currently active array (ex: 

). The way tag arrays are translated into colors can be set up using tag maps, also referred to as "Lookup tables" (LUT) or color transfer functions. Contrary to "scalar arrays" (see preceding section), tags are usually drawn manually with "painting tools".
For convenience purposes, as selected surfaces are drawn "grey" in MorphoDig, unselected surfaces objects can be tagged (tagging uniform "grey" objects without visual feedback would be uneasy). Tagging is the only way to modify an unselected surface in MorphoDig.
In order to be able to edit tags on a surface, the most common way is to create a new empty

tag array for this surface (see section 12.2 p. 12.2 for further details). You may read section 12.5 p.128 for a quick tagging starting guide.

12.1 Open Tags window

The "Tags" window can be also opened by clicking on "  " (see Fig. 12.1). The Tags window contains most options related to tags. The active tag array can be chosen here, as well as the active tag map. Tag maps (name, color, opacity of different biological structures) can be defined and modified here as well. The currently used tag tool is also chosen in this window (pencil "  ", paint bucket "  " or lasso "  ").

12.2 Create new empty tag array for each selected surface

This option will create a new "empty" tag array (the name of this new array is given by the user). "Empty", in this context, means that all vertices are assigned to the "Exterior" tag (tag=0).

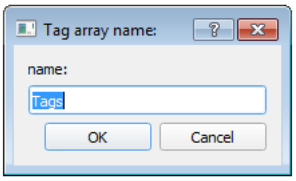


Figure 12.2: Creating a new tag array

12.3 Create new tag array based on currently displayed colors for each selected surface

Advice: **do not** use this method on surfaces displayed with **color gradients**. This option usually only makes sense when distinct colors are displayed (typically: less than 30).
This option (see Fig. 12.3) may be useful when you just have opened a .ply file already containing RGB colors (for instance, let us suppose that you have painted a surface using MeshLab software, and that you wish to convert those colors into tags). RGB colors contained in .ply files are inserted inside the RGB array when opened with MorphoDig. This option is not restricted to RGB arrays contained inside .ply files though: it is also possible to convert any displayed color (such as scalar color rendering via a discretized color table) into a tag value.

Two main ways to convert currently displayed colors are available :

1) Exact color match using active tag map

→ In that case, in order to be given a tag value other than Tag 00, a vertex must satisfy the following condition: its RGB scalar value should match one of the colors defined in the "Tags options" window for the active tag map. If a vertex does not satisfy this condition, it is given

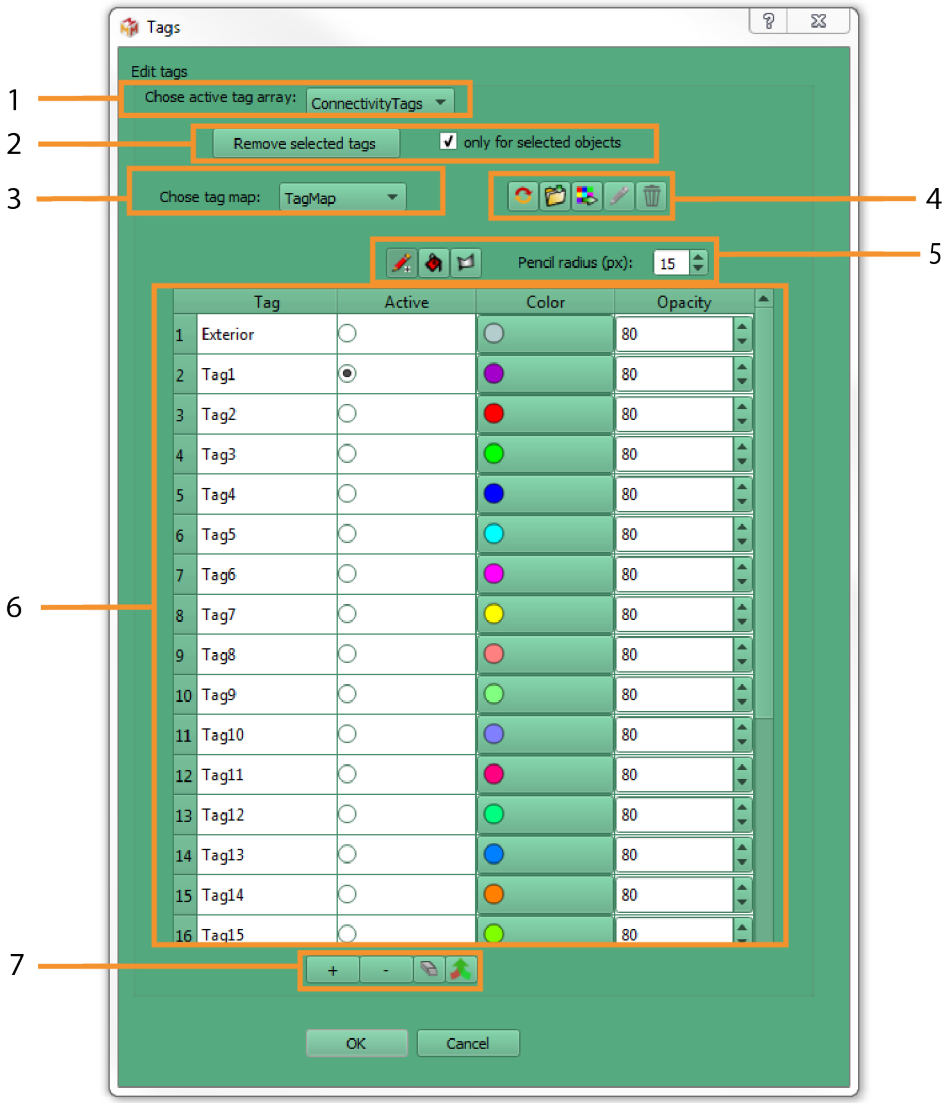


Figure 12.1: Tags window. This window is divided in different subsections. **1)** choose current active tag array. **2)** the active tag array can be deleted from selected/all surface objects in this sections. **3)** choose current active tag map, which transforms integer values into color and opacity on the screen. **4)** operations on the active tag map (from left to right). a: reinitialize tag map. b: Add tag map to presets = duplicate current active tag map and create a new custom tag map. c: export tag map(s) inside a .TAG or .TGP file. d: change active color map name (only possible for custom tag maps). d: delete active color map (only possible for custom tag maps). **5)** active tag tools (from left to right): a: pencil. b: paint bucket. c: lasso tag. d: pencil radius size (in pixels rendered on the screen). **6)** The tag map table: each line of this table is associated with an integer (exterior: 0; Tag1: 1 etc.), a color and an opacity. The active tag tools will paint a surface using the active tag. **7)** modification controls of the active tag map (from left to right). a: add a line to tag map table. b: remove last line from tag map table. c: reset active tag (merge with Tag 0 = Exterior). d: merge two tags.

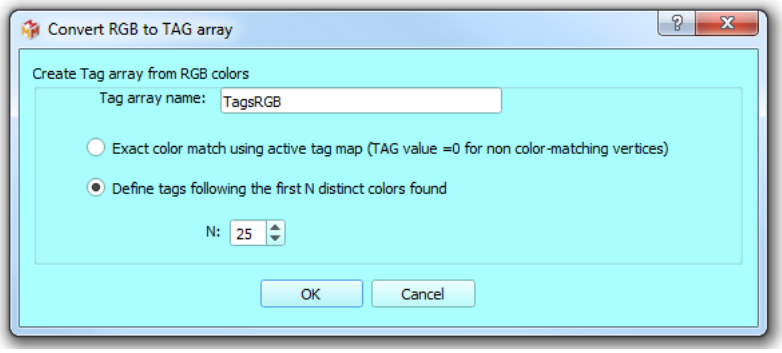


Figure 12.3: Create Tag array from RGB array.

the Tag 00 value (=exterior).

2) Define tags following the first N distinct colors found

→ In that case the following sequence of operations is performed:

a) Important: MorphoDig re-initializes the currently active tag map. If you have spent time to define colors and labels on the currently active tag map, it is strongly advised to save the active tag map inside a .TAG or .TGP file in order to be able to re-load it later. Otherwise, your work will be lost.

b) MorphoDig inspects the RGB display color of all the vertices of opened surfaces. When meeting a new color, if the active tag map contains less than "N" lines, MorphoDig inserts a new line inside the active tag map (of that color), and associates the current vertex id to this new tag id entry. All vertices of that same color will be associated to that id. If the color of the vertex is "new" and the maximum number of allowed entries in the tag map (N) has been reached, the vertex is associated with "tag id=0".

Method #1 will work perfectly if you are used to work with a well-defined set of colors on 3D surfaces, and have spent time to define a tag map with those colors.

Method #2 is much faster, but may lead to messy results: tag entries in the tag map will be ordered in a non-biological way.

Also remember not to use these methods when surfaces are displayed with color gradients. An example of use of method 2 is shown in Fig. 12.4 p.127.

12.4 Create new tag array based on connectivity for each selected surface

This option involves vtkPolyDataConnectivityFilter. This option will retrieve all non-connected regions of selected surfaces and assign to each of them a unique Tag id (see for instance Fig. 12.5 p.127).

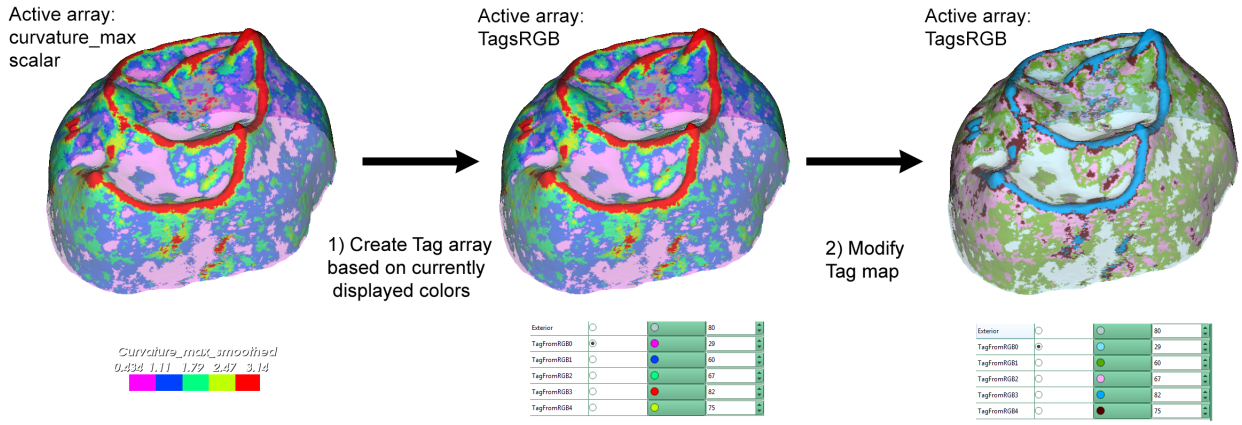


Figure 12.4: Example of automatic tag array creation based on displayed color of scalars. Note here that the scalar array (minimal curvature) is displayed using a **discretized color table** set up with a reasonably small number of colors (5 here). Specimen: enamel dentine junction (EDJ) of the second superior molar of a medieval human from Sains-en-Gohelle (France). Image credit: Mona Le Luyer.

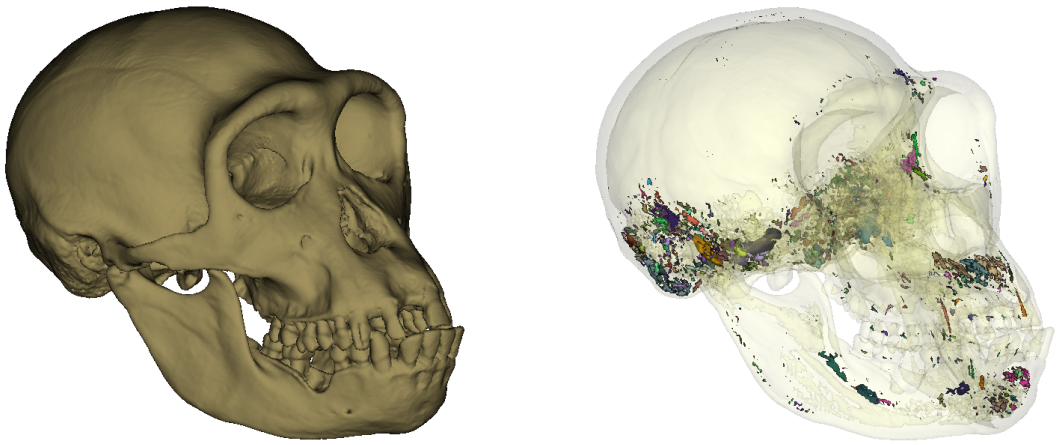


Figure 12.5: Example of tag array based on surface connectivity. Left: original surface representing the skull of the holotype specimen of *Pan paniscus*. Right: a tag array ware created based on the output of the connectivity procedure: a large large number (>1000) of non connected regions was found and each was tagged. The largest region (cranium and mandible) is displayed with an opacity of 10%, and the >1000 others with an opacity of 100%.

12.5 Tagging with MorphoDig: a quick starting guide

1: Make sure that your surface contains a Tag array: for instance, you may create a new empty tag array for this surface (see section 12.2 p. 12.2 for further details). Select this tag array as the currently active array.

2: Open the Tags window: a: if not already active, select desired active Tag. b: choose a tag tool (pencil "  ", paint bucket "  " or lasso "  ").

3: Tag in MorphoDig's main window.
a:when using the pencil "  " or the paint bucket "  ", you have two options:
"T" pressed +left click: allows color override
"T" pressed +right click: does not allow color override (see below for explanations of what color override is)
b: when using the lasso, maintain mouse left click pressed and draw a contour of the region which should be tagged.

12.5.1 Pencil tag tool

Pencil tag tool controls:
"T" pressed + left mouse click : tags the selected surface using currently active tag.
"T" pressed + right click : tags the selected surface using currently active tag **without color override**. Tag propagation will start at the picked vertex of a given color, but will stop when meeting another color different from that of Tag 0 (usually assigned to "exterior").

Pencil tag special option:
pencil tag size (in number of pixels) on the screen can be modified in the Tags window.

12.5.2 Paint bucket tag tool

Paint bucket tag tool controls:
"T" pressed + left mouse click : tags the selected surface using currently active tag.
"T" pressed + right click : tags the selected surface using currently active tag **without color override**. Tag propagation will start at the picked vertex of a given color, but will stop when meeting another color different from that of Tag 0 (usually assigned to "exterior").

When surfaces contain a lot of disconnected regions in space (for instance: the skeleton of

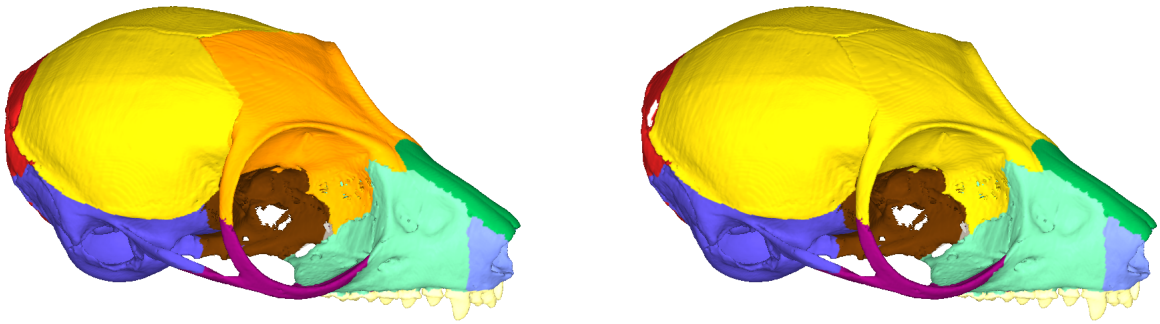


Figure 12.7: Example of tag merging. Left : cranium of *Microcebus murinus* presenting the parietal region tagged in yellow, the frontal region tagged in orange. Right : frontal tag region was merged into the parietal region.

a fetus), the paint bucket tool can be usefull to paint each disconnected element one by one in one mouse click. But be careful if your all regions of your surface are connected together: when using "T" + left click ("color override" allowed), as it will paint the whole object uniformly.

12.5.3 Lasso tag tool

Once the "lasso tag tool" button is pressed, draw the region to be tagged by dragging the mouse on the screen while maintaining the left button pressed. Then release the left mouse button.

12.5.4 Merge tags.

Two tagged regions can be merged into a single one. All source tags will be put into target tags. See for instance Fig. 12.7 p.129.

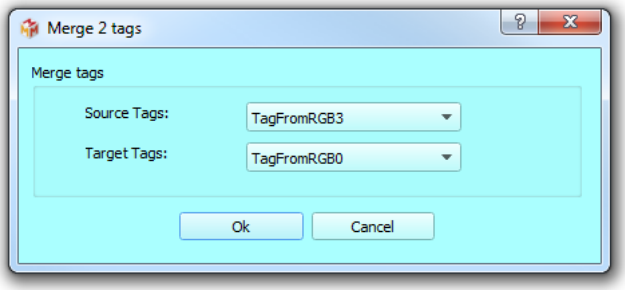


Figure 12.6: Merge tags window

Chapter 13

Menu RGB

Contents

13.1 Create or replace an RGB array with current display color

This option creates a new RGB array based on the currently displayed colors, depending on the currently displayed array / solid colors. If the newly created array is named exactly "RGB", surfaces can be exported inside .PLY file, and those colors can be visualized when such .PLY files with other software sucha as MeshLab.

Chapter 14

View

This menu contains the list of widgets which can be viewed or hidden as desired.

Chapter 15

Help

Contents

15.1 MorphoDig Web Site

This option opens http://morphomuseum.com inside a web navigator.

15.2 MorphoDig Tutorials

This option opens http://www.morphomuseum.com/tutorialsMorphoDig inside a web navigator.

15.3 About...

This window (see Fig. 15.1) shows the current version of MorphoDig.

Figure 15.1: About window.

Chapter 16

Acknowledgments

Contents

16.1 Design concepts

I wish to thank Christoph Zollikofer and Marcia Ponce de León, from whom I borrowed many of the design concepts implemented in FoRM-IT (Fossil Reconstruction and Morphometry Interactive Toolkit; [Zollikofer and Ponce de León, 1995, 2005]).

16.2 Code development

I wish to thank Stefan Schlager and Léo Botton-Divet for their helpful comments and their contribution to the cross-platform compatibility of the source code.

16.3 Specimens illustrated

I wish to thank the curators of different institutions for the access to most specimens illustrated in this document: Nathalie Mémoire from the Musée d'Histoire Naturelle de Bordeaux, Christoph Zollikofer and Marcia Ponce de León from the Anthropological Institute and Museum of Zürich, Suzane Jiquel from the ISEM. Thanks also to Mona Le Luyer and Pauline Colombet (PACEA, Bordeaux), the PEPS IdEx Bordeaux / CNRS 3 Dent'in for the access to

the digital reconstruction of the specimen SP07 of Sains-en-Gohelle, and Cédric Beauval and Archéosphère SARL (Bordeaux) for the access to these remains. Thanks to Lionel Hautier and Mark Wright for the access to the 3D model of nasal turbinates of *Orycteropus afer*

16.4 3D data acquisition facilities

I wish to thank the MRI Platform (Montpellier), Zollikofer and Marcia Ponce de León from the Anthropological Institute and Museum (Zürich), Paul Tafforeau and the E.S.R.F. (Grenoble, France) for the access to the scanning facilities in which the illustrated specimens were digitized.

References

Lisa S Avila, Sébastien Barré, Rustuy Blue, Diavid Cole, Berk Geveci, William A Hoffman, Brad King, C Chalres Law, Kenneth M Martin, William J Schroeder, and Amy H Squillacote. *The VTK User's Guide*. 2001. ISBN 1930934238. URL www.vtk.org.

Renaud Lebrun. MorphoDig, 2018. URL http://morphomuseum.com/morphodig.

Christoph P E Zollikofer and Marcia S. Ponce de León. Tools for rapid prototyping in the biosciences. *IEEE Computer Graphics and Applications*, 15(6):48–55, 1995. ISSN 02721716. doi: 10.1109/38.469515.

Christoph P E Zollikofer and Marcia S Ponce de León. *Virtual Reconstruction: A Primer in Computer-Assisted Paleontology and Biomedicine*. Wiley, New York, 2005. ISBN 978-0-471-20507-4.